Bádáme v kroužku informatiky na SŠ

Tento modul obsahuje náměty aktivit, které jsou vhodné pro realizaci v kroužku informatiky na střední škole. Jedná se o programování pro Android, vývoj aplikací zaměřených na animace a programování prostředí EV3 a Robot C se stavebnicemi Lego.

Obsah:

- ANDROID - programování v MIT AppInventoru
- ANDROID - programování v prostředí C#
- Programování prostředí EV3
- Programovací prostředí RobotC

Tento material vznikl z finanční podpory Evropského sociálního fondu a státního rozpočtu České republiky v rámci projektu „Popularizace vědy a badatelsky orientované výuky“, registrační čísloCZ.1.07/2.3.00/45.0007.

ANDROID

programování v MIT App Inventoru

Průvodce do světa pod povrchem Androidu. Kurz programování v blokově řízeném vývojovém prostředí MIT App Inventor.

Využité programy:

MIT App Inventor
App Inventor Emulátor
Emulátor Genymotion
Konektor aiStarter
Konektor MIT AI2 Companion
Android SDK

Cílová skupina/náročnost: 1. až 4. ročník SŠ a odpovídající ročníky gymnázií.

Autor:

Ing. Mgr. Filip Vaculík

Všechny uvedené texty, obrázky a videa jsou vlastní, není-li uvedeno jinak. Autory Youtube embed videí lze nalézt při kliknutí na znak Youtube ve videu během přehrávání.

K plnohodnotnému využití této studijní opory je nutný přístup k on-line zdrojům a materiálům.

Tento materiál vznikl z finanční podpory Evropského sociálního fondu a státního rozpočtu České republiky v rámci projektu „Popularizace vědy a badatelsky orientované výuky“, reg .č. CZ.1.07/2.3.00/45.0007.

1 Základní informace o projektu

Název: Programování pro Android v prostředí MIT App

Inventor Anotace programu/zaměření/hlavní cíl:

Seznámit žáky s atraktivní možností tvorby aplikací pro Android, přímo ve webovém prohlížeči. Kromě seznámení s prostředím Androidu může být vývojové prostředí App Inventor i vhodným nástrojem k výuce algoritmizace a úvodem do programování. Podpora a přímé využití principů 1:1 a BYOD.

Cílová skupina

Technicky orientovaní žáci ZŠ, žáci gymnázií a SŠ, účastníci kurzů programování pro začátečníky.

Organizační podmínky

Předpokládá se výuka v počítačové učebně s menším počtem žáků (cca 8 až 15). V případě distančního vzdělávání lze formy a metody výuky přizpůsobit.

Pomůcky
PC s parametry:

CPU 1,5GHz, 1GB RAM, doporučované prohlížeče Mozilla Firefox nebo Google Chrome v aktuální verzi.
Operační systém není rozhodující, podporovány jsou nejen hlavní 3 OS (Win, Lin, Mac).

Výhodou je vlastní zařízení s OS Android a WiFi připojení k internetu.

V případě použití emulátoru nároky na hardware PC vzrůstají dle typu a požadavků emulátoru.

Časová náročnost (popř. jak je možné program rozložit – jedná- li se o celoroční program)

Kurz je dimenzován na 12 dvouhodinových lekcí. Další čas může být věnován volitelným činnostem, které dle časových možností a zájmu žáků mohou i výrazně překročit původní záměr. Podporu pro vedení volitelných čiností lze nalézt v materiálech mimo hlavní vzdělávací blok.

Mezipředmětové vazby

Vzhledem k prostoru pro volitelné směrování tvorby programů lze do výuky zahrnout mezipředmětové vazby pro jakýkoliv předmět. Charakterem práce se nelze vyhnout mezipředmětovým vazbám na anglický jazyk, matematiku, český jazyk a ICT předměty.

2 Motivační rámec projektu

Text:

Elektronická zařízení se velmi rychle stala významnou součástí osobního i profesního života většiny populace. Působení těchto zařízení ovlivňuje myšlení a kulturu v nebývalém měřítku. Motivační potenciál této oblasti je obrovský. Zde stojíme na hranici mezi těmi, kdo vytváří, a těmi, kdo pouze konzumují multimediální obsah. Snad vás tento kurz posune dále do světa tvůrců a přinese vám lepší pochopení technologií, které ovlivňují váš život.

Doporučený multimediální materiál:

video viz. on-line kurz

3 Poznámky k využití programů

Text:

Samotný App Inventor je díky svému návrhu poměrně nenáročný. Studenti budou potřebovat vlastní Google účet, případně účet Google for education, pokud jej má škola zřízený. Propojení s vlastním zařízením s Androidem je nenáročné a postup lze nalézt na stránce http://appinventor.mit.edu/explore/ai2/setup.html. Problematičtější může být zprovoznění emulátorů v případě, kdy žáci nemají vlastní zařízení, případně není dostupné kvalitní bezdrátové připojení pro všechna zařízení žáků.

Zprovozňování emulátorů je v tomto kurzu zpracováno do samostatných volitelných bloků.

Doporučený multimediální materiál:

App Inventor je prostředí, které používají statisíce uživatelů po celém světě. Mnoho materiálů lze nalézt na stránce http://appinventor.mit.edu/explore/.Pro práci s tímto prostředím existuje řada ukázkových videí, dostupných například na YouTube.

Některá z videí jsou použita i v tomto kurzu. Stále vznikají nové materiály, tento kurz je toho příkladem. Pokud najdete nový zajímavý materiál, napište nám prosím o něm do wikipedie našeho kurzu.

4 Aktivita 1 - seznámení s prostředím a Hello World

Téma	Programování pro Android v prostředí MIT App Inventor	
Tematický celek	Seznámení s prostředím App Inventoru	
Motivační rámec	Snadné vytvoření aplikace pro Android a její použití.	
Počet žáků	5-15	
Věk studentů	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/, základní orientace v prostředí, tvorba první aplikace, propojení se zařízením s Androidem/ s emulátorem, spuštění vlastní aplikace.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti budou s dopomocí učitele schopni splnit jednotlivé cíle, a pokud se nesetkají s technickými obtížemi, budou schopni tento postup zopakovat v jiném prostředí, například doma.	
Předchozí znalosti	Znalost práce s webovým prohlížečem. Orientace v prostředí komplexní webové aplikace. Základní znalost anglického jazyka.	
Mezipředmětové vztahy	Anglický jazyk, IT předměty.	
Časový plán	Fáze činnosti	Metody a formy, motivace
volitelně 5m	Seznámení a ukázka práce s Moodle.	Frontální výklad, ukázka
volitelně 10m	Přihlášení studentů k LMS Moodle, orientace v prostředí, vyplnění vstupního dotazníku.	Samostatná práce s dopomocí
volitelně 10m	Ukázka založení účtu pro Google služby. Založení účtů studenty.	Ukázka a samostatná práce s dopomocí
5m	Ukázka přihlášení k prostředí App Inventoru a základní orientace v prostředí. Ukázka vytvoření prvního programu, ukázka funkčnosti prvního programu na zařízení nebo v emulátoru.	Frontální výklad, ukázka, diskuze
10m	Přihlášení studentů do prostředí App Inventoru, tvorba prvního programu, připojení zařízení nebo emulátoru, spuštění programu.	Samostatná práce s dopomocí, zhodnocení a diskuze
10-45m	Modifikace programu, řešení individuálních potřeb, průběžná diskuze a spolupráce mezi studenty.	Samostatná a skupinová práce
15m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje srovnání cílů aktivity a vlastního postupu. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují znalost prostředí App Inventoru.	
Poznámky	Úspěšnost velmi závisí na přípravě prostředí. Doporučuji nepodceňovat přípravu a dopředu připravit a otestovat funkčnost emulátorů, případně schopnost WiFi obsloužit najednou všechna zařízení studentů, připravit záložní Google účty. Pro případ výpadku připojení k internetu doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Doporučený multimediální materiál

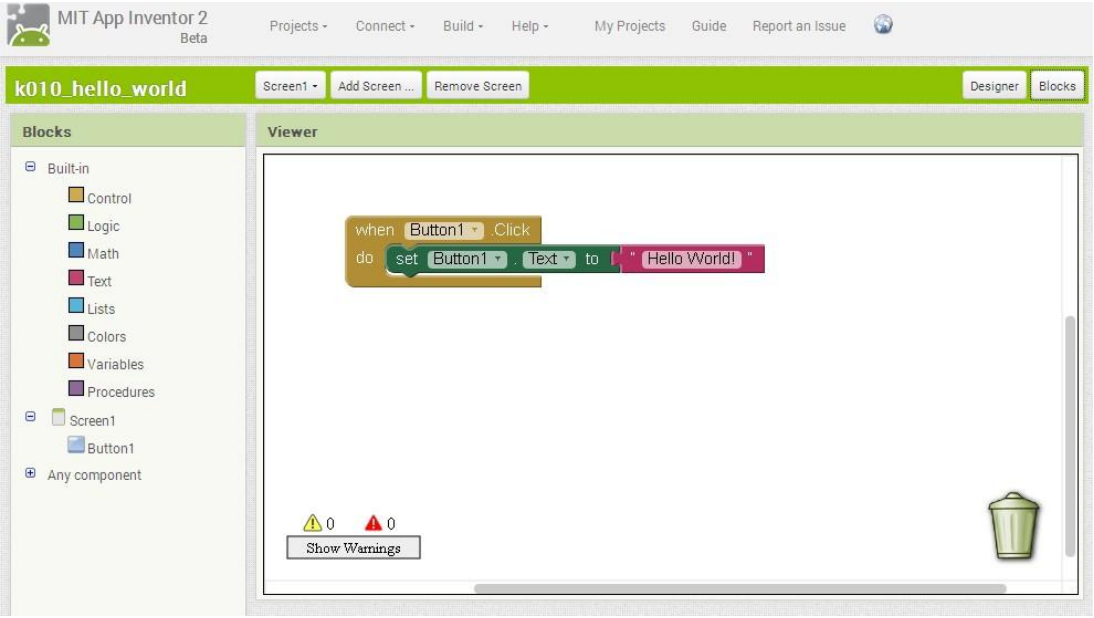
PDF tutoriál k seznámení s prostředím App Inventoru a vytvoření první aplikace.

Video pro seznámení s aplikací a tvorbou první aplikace. Doporučuji však provést vlastní ukázku.

(materiály viz. on-line kurz)

Ukázka aplikace Hello World

Účelem je předvést snadnost vytvoření aplikace a ověřit propojení App Inventoru se zařízením studenta nebo s emulátorem.

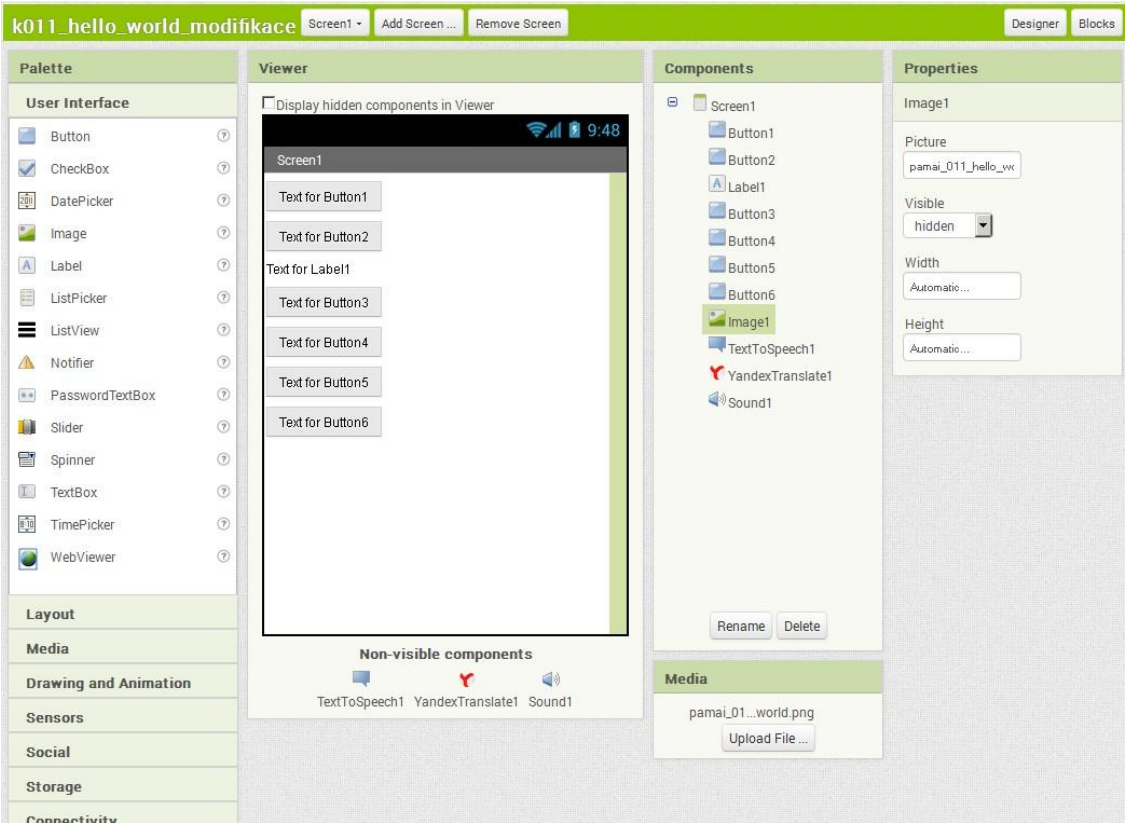


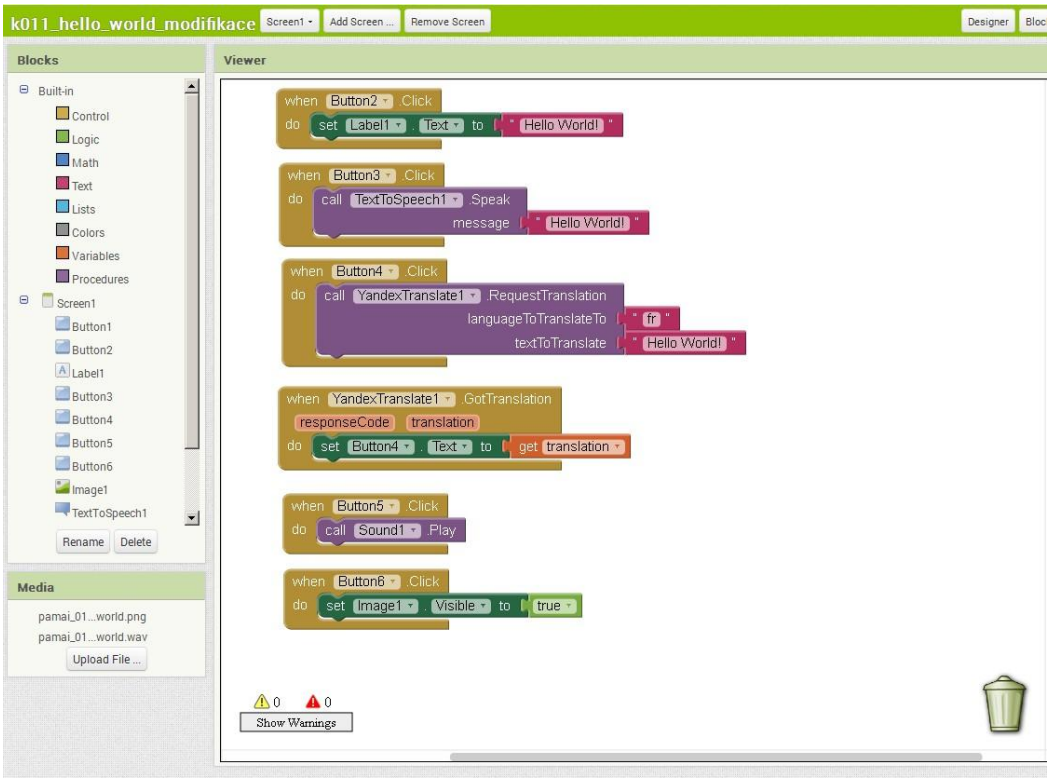
Modifikace první aplikace

V části modifikací, může být studentům doporučeno několik úprav, ve kterých se opakuje námět "Hello World" v různých modifikacích. Účelem této části lekce je dát prostor studentům, aby se zamysleli nad možnosti, které jim App Inventor přináší, a vyzkoušeli, nakolik si s tím prostředím rozumí.

V ukázce je obrazovka designeru a bloků programu, který je rozšířen o další funkce. Podobně jako v základním programu, i zde je každá akce spuštěna kliknutím na tlačítko.

- Tlačítko 2 vyvolá akci přejmenování nápisu v komponentě Label1.
- Tlačítko 3 provede strojové čtení textu "Hello World!". Komponenta TextToSpeech je vložena v designeru ze sekce Media. Postup je zobrazen v dříve zmíněném ukázkovém videu video se seznámením. (viz. on-line kurz)
- Tlačítko 4 spustí akci překladu textu. Text "Hello World!" je přeložen pomocí komponenty YandexTransle opět v designeru ze sekce Media. Po přeložení je zavolána akce when yandexTranslate1.GotTranslation a přeložený text je vložen do názvu tlačítka
  4. Popis práce s touto komponentou je na videu s ukázkou práce s YandexTranslate. (viz. on-line kurz)
- Tlačítko 5 provede akci přehrátí zvuku "Hello World". Tato akce, stejně jako akce tlačítka 3, vyžaduje, aby zařízení nebo emulátor správně pracovaly se zvukem.
- Tlačítko 6 zobrazí skrytý obrázek s textem "Hello World". Obrázek byl vložen pomocí komponenty Image ze sekce User Interface, následně byl uploadován pomocí vlastnosti Picture (sloupec vpravo) a poslední úprava byla skrytí obrázku pomocí vlastnosti Visible hidden.


k011_hello_world_modifikace
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Palette
User Interface
Button
CheckBox
DatePicker
Image
Label
ListPicker
ListView
Notifier
PasswordTextBox
Slider
Spinner
TextBox
TimePicker
WebViewer
Layout
Media
Drawing and Animation
Sensors
Social
Storage
Viewer
Display hidden components in Viewer
9:48
Screen1
Text for Button1
Text for Button2
Text for Label1
Text for Button3
Text for Button4
Text for Button5
Text for Button6
Non-visible components
TextToSpeech1
YandexTranslate1
Sound1
Components
Screen1
Button1
Button2
Label1
Button3
Button4
Button5
Button6
Image1
TextToSpeech1
YandexTranslate1
Sound1
Rename
Delete
Media
pamai_01...world.png
Upload File ...
Properties
Image1
Picture
pamai_011_hello_wo
Visible
hidden
Width
Automatic...
Height
Automatic...


k011_hello_world_modifikace
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Viewer
Built-in
Control
Logic
Math
Text
Lists
Colors
Variables
Procedures
Screen1
Button1
Button2
Label1
Button3
Button4
Button5
Button6
Image1
TextToSpeech1
Rename
Delete
Media
pamai_01...world.png
pamai_01...world.wav
Upload File ...
when Button2 .Click
do set Label1 . Text to " Hello World! "
when Button3 .Click
do call TextToSpeech1 .Speak
message " Hello World! "
when Button4 .Click
do call YandexTranslate1 .RequestTranslation
languageToTranslateTo " fr "
textToTranslate " Hello World! "
when YandexTranslate1 .GotTranslation
responseCode translation
do set Button4 . Text to get translation
when Button5 .Click
do call Sound1 .Play
when Button6 .Click
do set Image1 . Visible to true
0
0
Show Warnings

5 Aktivita 2 - aplikace využívající kreslící plochu "canvas"

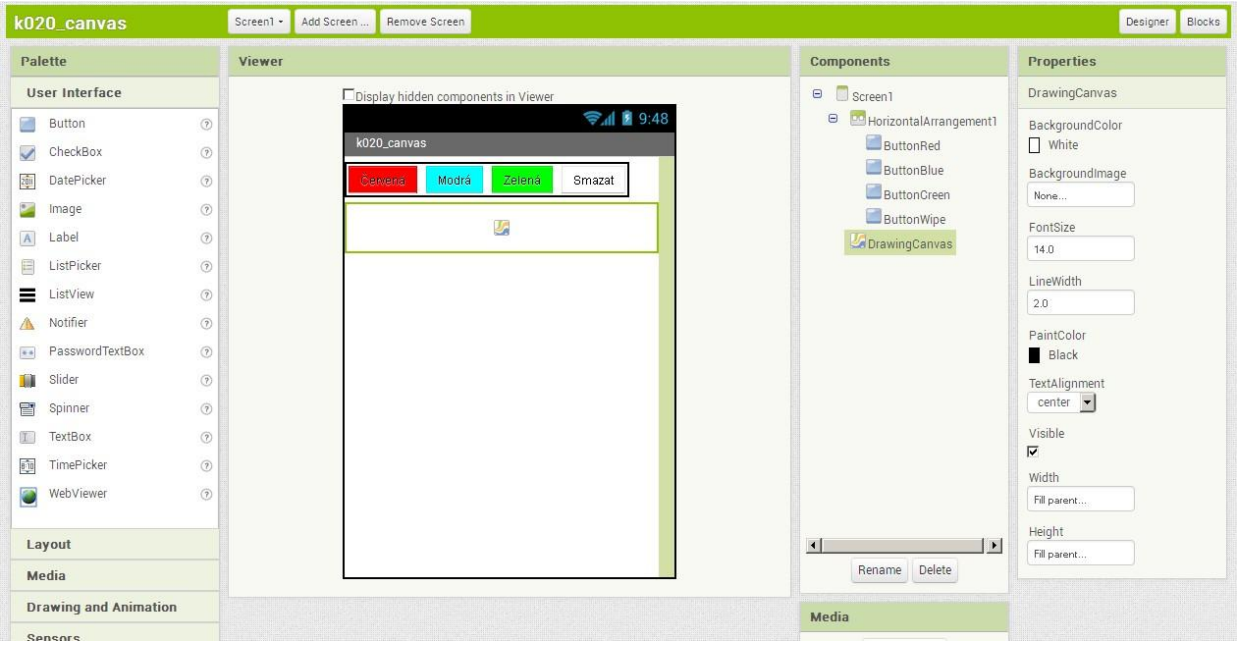
Tematický celek	Kreslení do komponenty kreslicí plocha "canvas"	
Motivační rámec	Tvorba vlastní grafické aplikace s vizuálními efekty	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/. Tvorba, testy a ladění aplikace v online vývojovém prostředí.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti zvládnou vytvoření jednoduchého grafického editoru, jehož funkce mohou dále rozvíjet.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
volitelně 5-10m	Zopakování základních informací o prostředí AppInventoru. Ukázka prostředí a popis ovládání.	Frontální výklad, ukázka
5m	Seznámení s tématem lekce. Volitelně ukázka funkční aplikace. Seznámení s výukovými materiály.	Frontální výklad, ukázka
25m	Tvorba aplikace dle tutoriálu a videonávodu.	Samostatná práce s využitím výukových materiálů
15-45m	Modifikace aplikace	Samostatná a skupinová práce
15m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují znalost prostředí V App Inventoru.	
Poznámky	V případě, že by použití odkazovaných výukových zdrojů v anglickém jazyce činilo studentům problémy, je doporučen postup projít s nimi krok za krokem základní aplikaci a komentovat jednotlivé kroky. Pro případ výpadku připojení k internetu doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Doporučený multimediální materiál

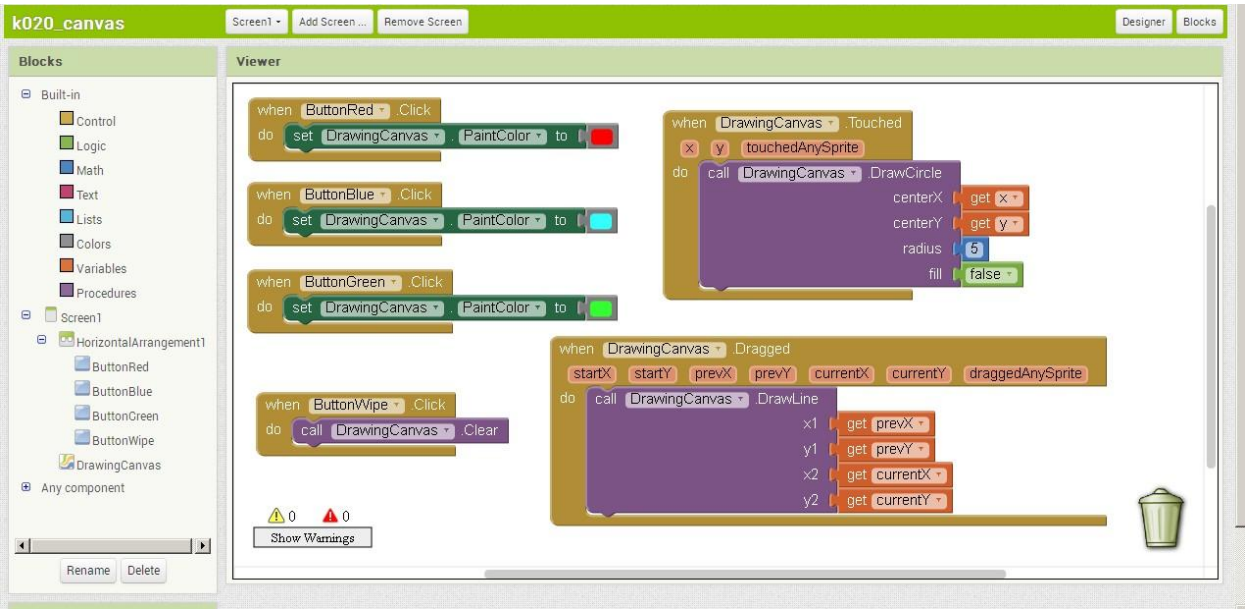
Tutoriál věnující se práci s kreslicí plochou (canvas). Základní aplikace se provedením záměrně blíží aplikaci v tutoriálu. Některé prvky, jako například obrázek na pozadí, jsou z důvodu vyšší přehlednosti vynechány. Uživatelské prostředí aplikace je v českém jazyce, identifikátory objektů v programu jsou pojmenovány v anglickém jazyce pro zachování shody s tutoriálem. (viz. on-line kurz)

Aplikace pro práci s kreslicí plochou "canvas"

Ukázka základní aplikace pro práci s kreslicí plochou, design.



Ukázka kódu základní aplikace pro práci s kreslicí plochou.



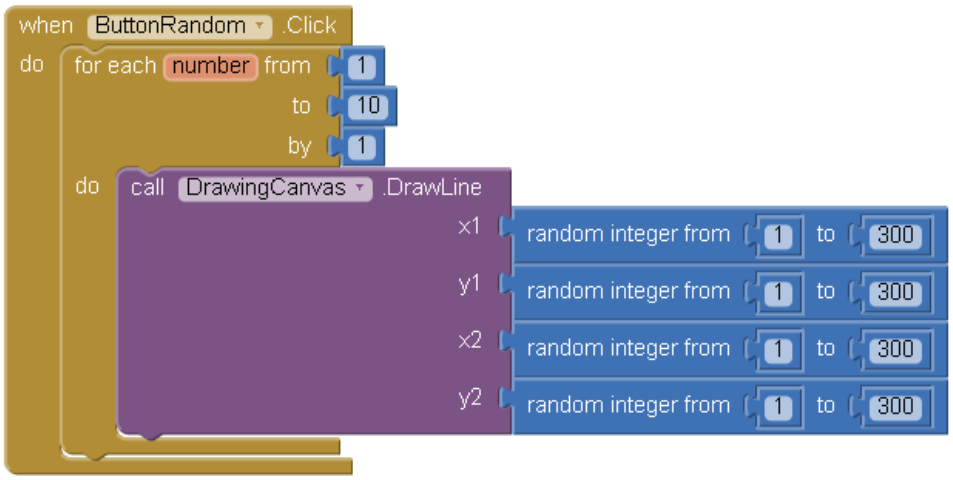
Modifikace základní aplikace

Studenti mohou být ponecháni v experimentování s kreslicí plochou a kombinováním znalostí z předchozí lekce, případně mohou být nasměrováni ke konkrétním modifikacím.

Např:

- doplnění obrázku na pozadí, přidání více barev,
- generování náhodné kresby **viz obrázek níže,** kresba různých geometrických tvarů,
- zvětšování a zmenšování kreslených geometrických tvarů a čáry, namíchání náhodné barvy,
- uložení obrázku do souboru,
- další náměty lze nalézt ve sborníku kódu https://puravidaapps.com/snippets1.php#canvas.

Ukázka kódu generujícího náhodné čáry.



6 Aktivita 3 - aplikace používající grafické objekty "sprites"

Tematický celek	Manipulace s grafickými objekty "sprity"	
Motivační rámec	Tvorba interaktivní grafické aplikace	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Tvorba, testy a ladění aplikace v online vývojovém prostředí.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti zvládnou vytvoření interaktivní grafické aplikace.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-10m	Seznámení s tématem lekce. Volitelně ukázka funkční aplikace. Seznámení s výukovými materiály.	Frontální výklad, ukázka
5-10m	Volitelně informace k tvorbě obrázků a průhlednosti. Volitelně ukázka práce s grafickým editorem, export a upload. Volitelně popis kolizí grafických objektů.	Frontální výklad, ukázka
35m	Tvorba aplikace dle tutoriálu a videonávodu.	Samostatná práce s využitím výukových materiálů
15-45m	Modifikace aplikace	Samostatná a skupinová práce
15m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	V případě, že by použití odkazovaných výukových zdrojů v anglickém jazyce činilo studentům problémy, je doporučen postup projít s nimi krok za krokem základní aplikaci a komentovat jednotlivé kroky. Pro případ výpadku připojení k internetu doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

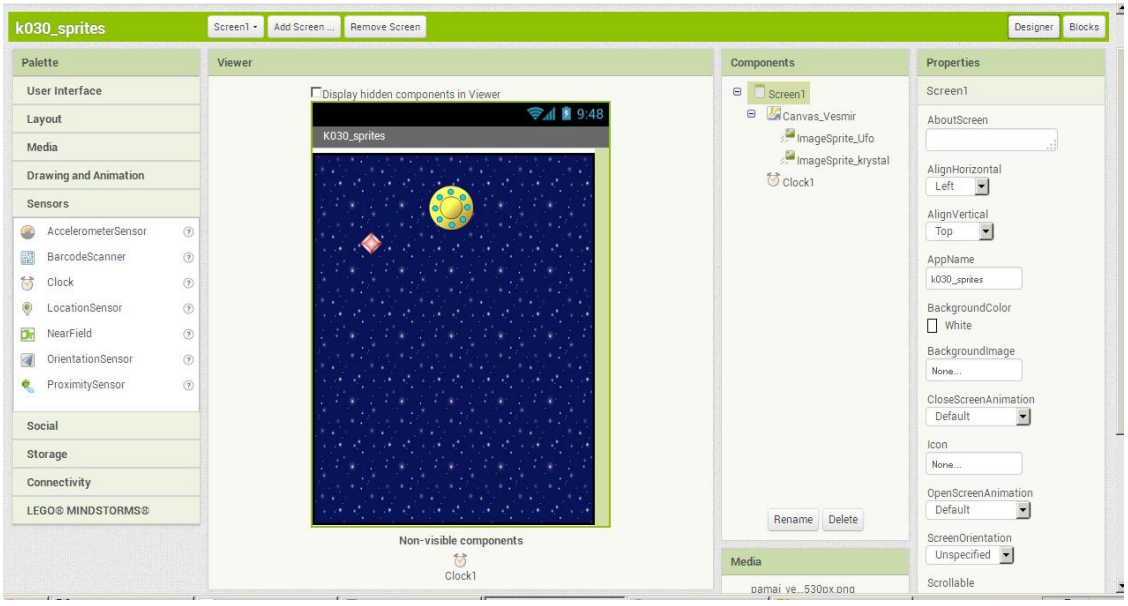
Doporučený multimediální materiál

Tutoriál provádějící užitím grafických objektů, sprites. (viz. on-line kurz)

Aplikace pro ukázku práce s grafickými objekty

Ukázka modelové aplikace.

TIP: Pohyb sprite se řídí z designeru, při vybraném sprite, hodnotou Speed v menu Properties. Grafické soubory naleznete v sekci Grafické soubory použité v kurzu.



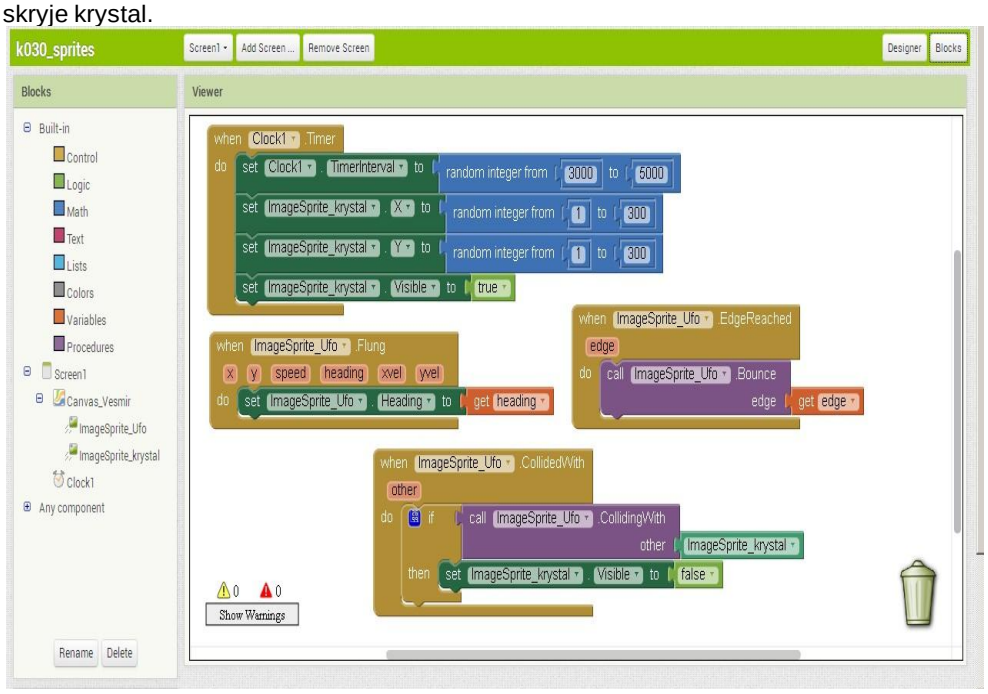
Ukázka kódu základní aplikace pro práci s grafickými objekty

Bloky when Clock1.timer = dle nastaveného intervalu provádí v náhodném čase 3 až 5 sekund přesun krystalu, jeho součástí je zviditelnění krystalu, pro případ, že by byl skryt.

When ImageSprite_Ufo.Flung = zajišťuje změnu směru sprite určenou na základě přejetí prstem.

When ImageSprite_Ufo.EdgeReached = zajišťuje akci v případě, kdy se UFO dostane k okraji plochy.

When ImageSprite_Ufo.ColideWith = obsluha akce kolize s jiným sprite. V případě, že je kolidujícím objektem krystal, skryje krystal.



Modifikace základní aplikace

Studenti mohou být ponecháni v experimentování s grafickými objekty a kombinováním znalostí z předchozí lekce, případně mohou být nasměrování ke konkrétním modifikacím.

Např:

doplnění více sprites,
export 3D sprites ze zdrojového grafického souboru, tvorba vlastních obrázků pro sprites,
změna chování sprites při detekci kolize meteory jako překážky.

7 Aktivita 4 - použití proměnných "variable"

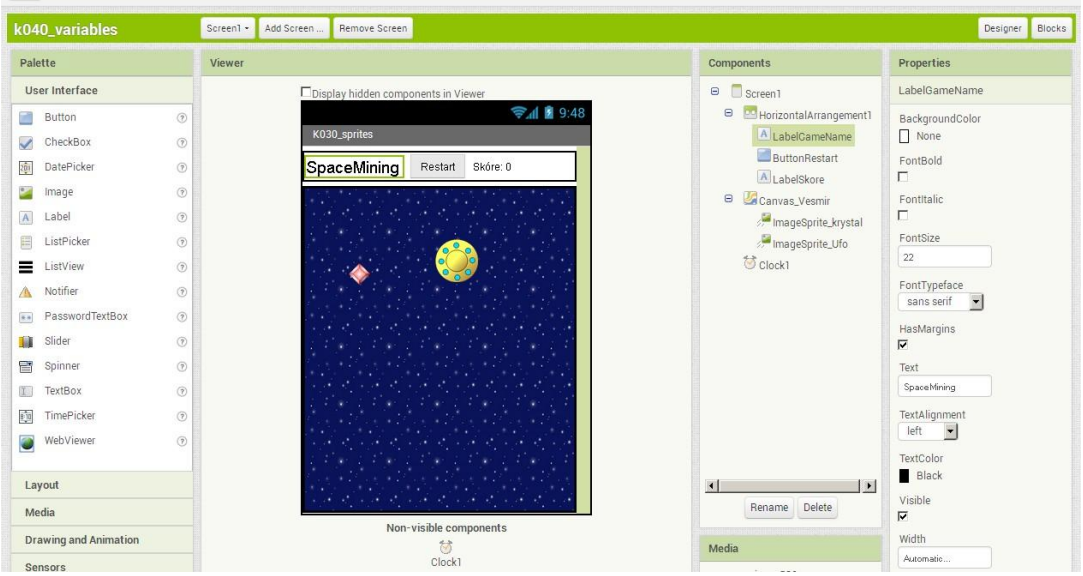
Tematický celek	Použití proměnných "variables"	
Motivační rámec	Rozšíření funkcí interaktivní grafické aplikace.	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Tvorba, testy a ladění aplikace v online vývojovém prostředí.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti si vyzkouší použití proměnných, jejich inicializaci, nastavování a změnu hodnot, používání proměnných.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-10m	Seznámení s tématem lekce. Volitelně ukázka funkční aplikace. Seznámení s tématem proměnných. Globální vs lokální proměnné.	Frontální výklad, ukázka
25m	Tvorba aplikace dle ukázek a popisu v této lekci.	Samostatná práce s využitím výukových materiálů tohoto kurzu
15-60m	Modifikace aplikace. Volitelně, inspirace v tutoriálu hry	Samostatná a skupinová práce
15m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postupu a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Doporučený multimediální materiál

Tutoriál ukazující jinou techniku práce s proměnnou, využívající hodnotu v komponentě label. (viz. on-line kurz)

Aplikace pro ukázku práce s proměnnými

Zde proti předchozí verzi přibylo několik komponent. Pomocí Layout manageru jsou uspořádány popisek s názvem hry, tlačítko pro restart hry a popisek, pomocí kterého bude zobrazováno skóre hry.

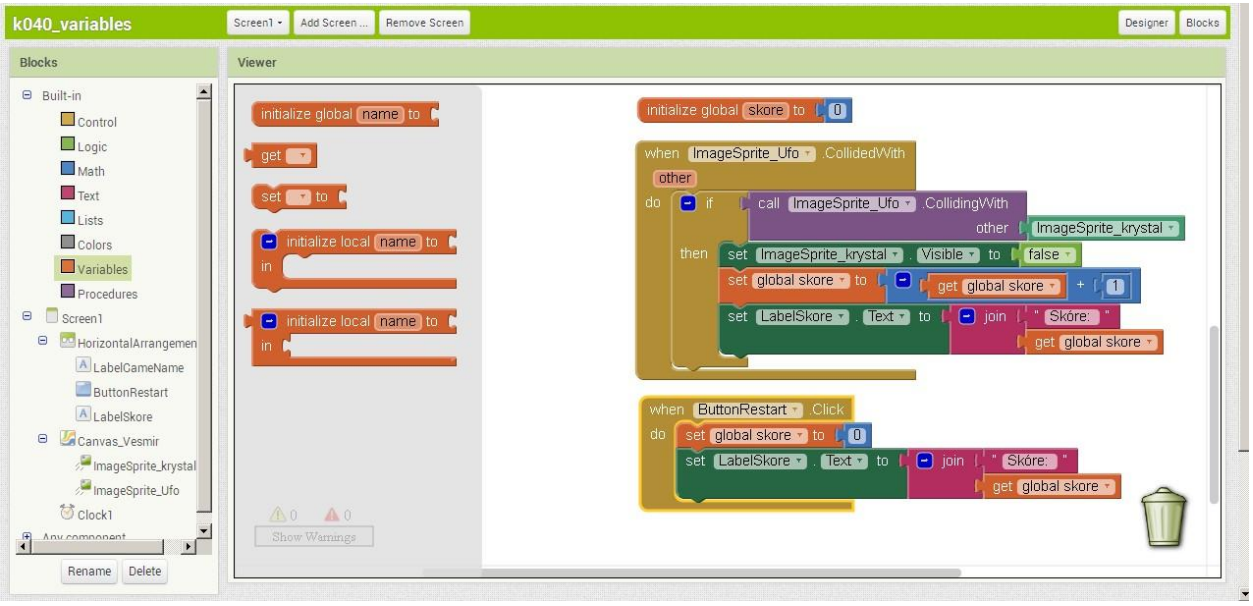


Ukázka kódu základní aplikace pro práci s proměnnými Bloky:

Initialize global skore to 0 = výchozí inicializace globální proměnné, ke které lze přistupovat z libovolných bloků programu.

When ImageSprite_Ufo.ColideWith = blok obsluhy kolize s jiným sprite byl rozšířen a při detekci kolize je navyšována hodnota v promenné skore o 1 a nová hodnota je zobrazována v LabelSkore.

When ButtonRestart Click = provede akci vynulování skóre po kliknutí na tlačítko reset.



Modifikace základní aplikace

Doporučené modifikace programu:

> doplnění více sprites krystalů a přičítání různého skóre pro různé krystaly, vytvoření proměnné pro počet životů a odečítání životů při srážce s meteoridem, vytvoření proměnné pro rychlost UFA a změna rychlosti při sebrání "vylepšení", zrychlení jen na omezený čas, nastavení původní rychlosti po restartu.

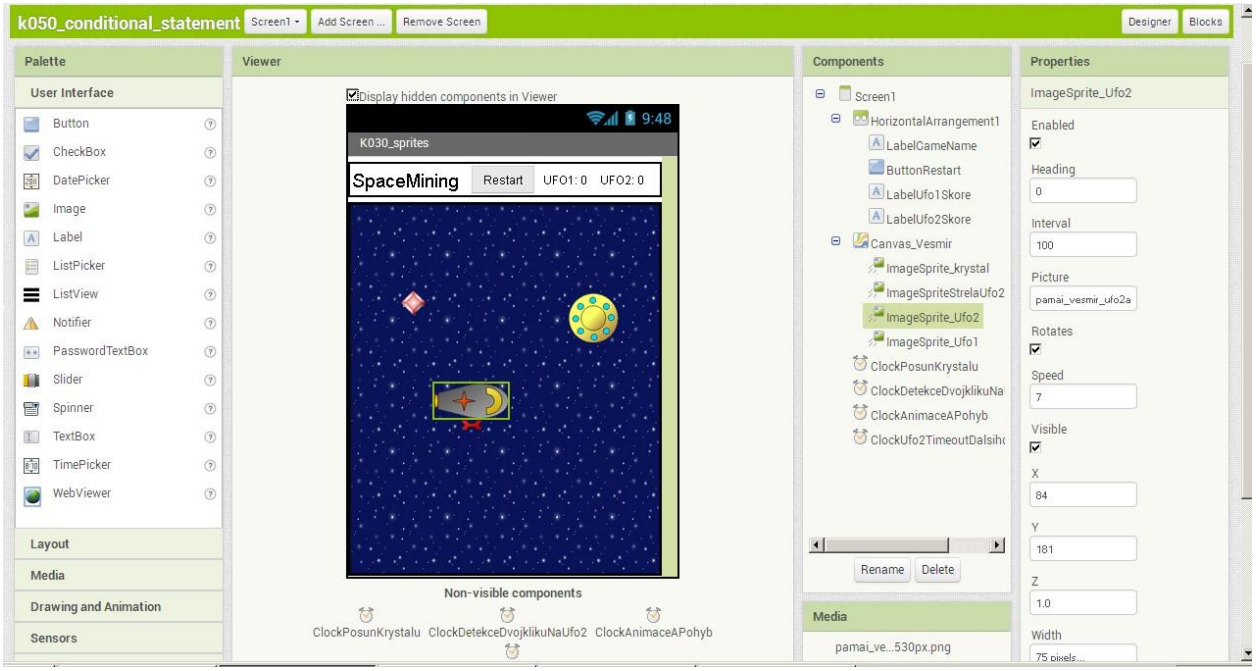
8 Aktivita 5 - podmíněné větvení programu

Tematický celek	Použití podmíněného větvení programu	
Motivační rámec	Multiplayer v interaktivní grafické aplikaci	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Tvorba, testy a ladění aplikace v online vývojovém prostředí.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti si vyzkouší použití několika způsobů větvení programu.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-25m	Seznámení s tématem lekce. Volitelně ukázka funkční aplikace. Seznámení s tématem větvení programu. Neúplné podmíněné větvení "if", úplné podmíněné větvení "if / else", vícenásobné podmíněné větvení "if / else if / else". Volitelně nákres vývojových diagramů. Volitelně samostatná činnost žáků, vyhledávání informací k tomuto tématu.	Frontální výklad, ukázka, samostatná práce žáků
30-60m	Úprava vlastní aplikace dle ukázek a popisu v této lekci.	Samostatná a skupinová práce s využitím výukových materiálů tohoto kurzu
15-20m	Modifikace aplikace.	Samostatná a skupinová práce
15m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Video s tipem, jak je možné pracovat s animacemi. (viz. on-line kurz)

Aplikace pro ukázku podmíněného větvením programu

Opět přibylo několik komponent, především několik časovačů, loď pro druhého hráče, střela.



Ukázka části kódu aplikace, ve které je využito několik typů podmíněného větvení programu. Nově přibyla

funkčnost:

loď druhého hráče a obsluha jejího pohybu,
obsluha výstřelu při dvojkliku na loď druhého hráče, z pozice x:y lodi ve směru letu lodi, animace letící střely,
animace motorů lodi druhého hráče,
obsluha kolize střely druhého hráče s lodí prvního hráče, zastavení lodi prvního hráče po zásahu střelou druhého hráče, obsluha kolize lodi druhého hráče s krystalem a počítání skóre,
obnova nastavení pro střelu a druhého hráče při stisku tlačítka reset, timeout zabraňující opakování výstřelů rychle za sebou.

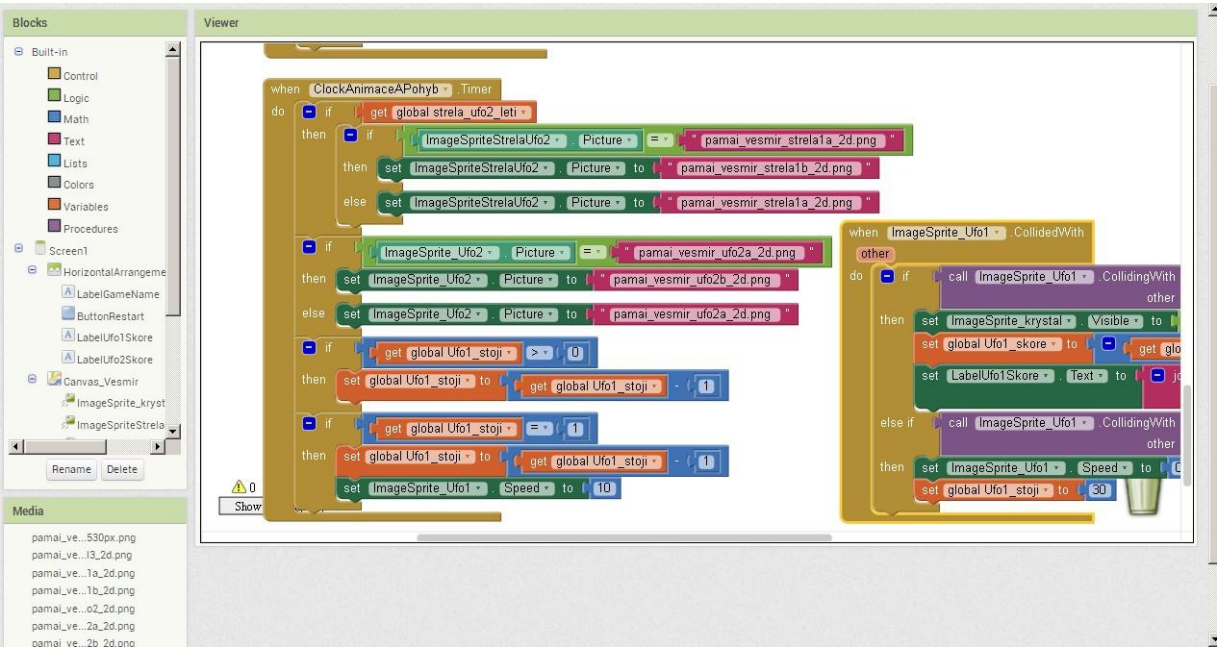
Významné bloky:

when ClockAnimaceAPohyb.Timer = časovač tikající po 100ms, řešící animaci (změnu obrázků) střely, v případě, že je letící. Animace motoru lodi druhého hráče. Řeší i zastavení lodi prvního hráče po zásahu střelou. Zastavení je řízeno proměnnou Ufo1_stoji, která je při zásahu nastavena na hodnotu 30. Loď stojí po dobu cca 30x 100Ms.

When ImageSprite_Ufo.ColideWith = V bloku kolize je navíc řešena obsluha zásahu střelou.

When ImageSprite_Ufo2.Touched = detekce dvojkliku na loď druhého hráče. V totmto případě je snaha o odfiltrování dotyku, který nastává při ovládání směru pohybu lodi. Za dvojklik je považován dvojdotek lodi, který nastane v čase menším než 300Ms.

When ClockDetekceDvojklikuNaUfo2 = pomocný blok, který pomáhá při detekci dvojkliku.



Modifikace základní aplikace

Doporučené modifikace programu:

možnost střelby i pro prvního hráče, druhý hráč reagující na zásah, animace střelby prvního hráče, animace lodi prvního hráče,
detekce kolize lodí hráčů a přenesení lodí na protilehlé strany plochy, životy hráčů a odečítání při zásazích,
zastavení a vyhodnocení hry po určitém časovém intervalu / po dosažení počtu bodů.

9 Aktivita 6 - cykly a procedury jako řídící struktury programu

Tematický celek	Použití cyklů a procedur jako dalších řídicích struktur programu	
Motivační rámec	Seznámení s důležitým aspektem programování s širokým využitím při vlastní tvorbě.	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Teoretické a praktické seznámení s cykly programu, které jsou k dispozici v programu App Inventor.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti se seznámí s cykly programu a vyzkouší si jejich použití.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-35m	Seznámení s tématem lekce. Dle znalostí studentů seznámenís cykly a procedurami nebo zopakování cyklů a procedur jako dalších struktur pro řízení běhu programu. Seznámení s pojmy/zopakování pojmů jako: nekonečný cyklus/smyčka, cyklus s podmínkou na začátku, cyklus s podmínkou na konci, počítaný cyklus. Volitelně nákres vývojových diagramů. Vysvětlení / připomenutí pojmů procedur / funkcí, jejich definice, volání vstupních a výstupních parametrů.	Frontální výklad, ukázka, samostatná práce žáků
30-60m	Využití znalosti cyklů a procedur při tvorbě vlastní aplikace.	Samostatná a skupinová práce s využitím získaných znalostí, infromací v tomto kurzu a externích zdrojů.
15-20m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Wikipedie k tématu řídících struktur. (odkaz viz. on-line kurz)

Doplňkový tutoriál na aplikaci Xylofón. Aplikace využívá procedury.

Aplikace pro ukázku cyklů a procedur

Zásadní věc je, že App Inventor vytváří jednovláknové aplikace a jakýkoliv blok programu, pokud je spuštěn, je vykonán až do konce. To je třeba mít na paměti, obzvláště pokud jste zvyklí na vícevláknové aplikace. **V programech App Inventoru se každý blok, který své ukončení podmiňuje změnou na jiném místě programu, stává nekonečnou smyčkou.** Z toho důvodu není v App Inventoru implementována funkce pozastavení běhu (sleep / pause / delay) a opakující se akce v určitých intervalech jsou řešeny pomocí časovačů.

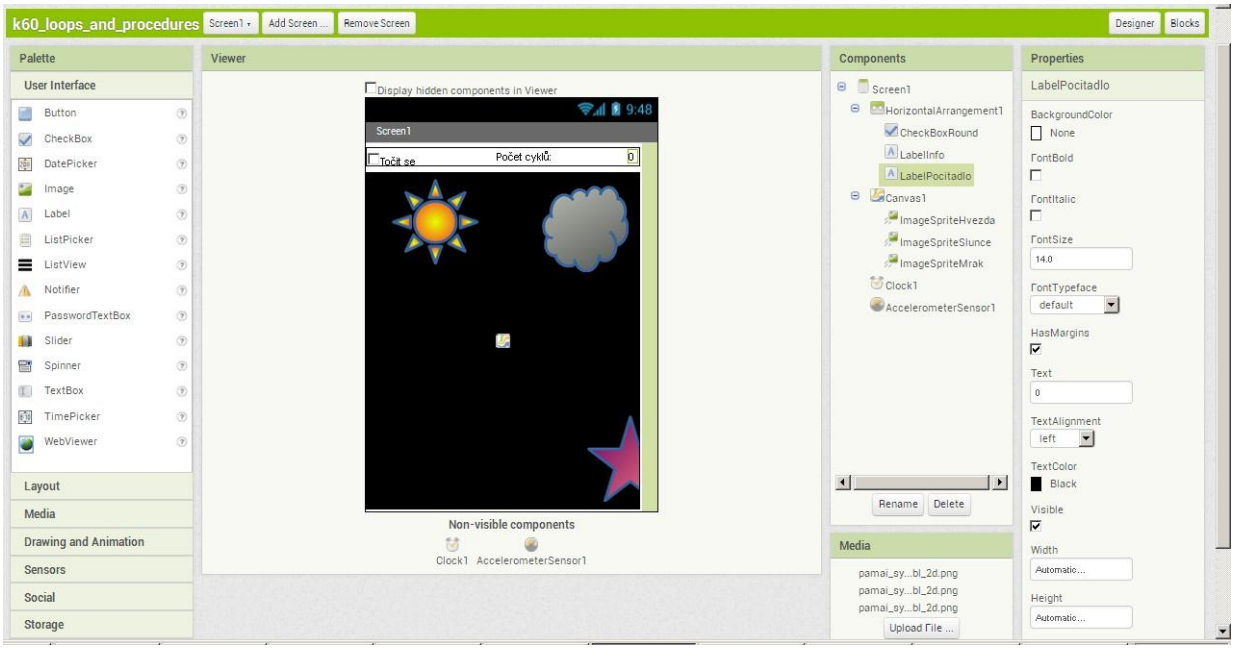
Více o této vlastnosti programu lze naléz na blogu Jose Florese. (odkaz viz. on-line kurz)

Cyklus "while test" lze používat pro akce, které mají konec uvnitř svého cyklu.

Z pohledu cyklů lze smyčky s podmínkou na začátku nahradit časovači s TimerEnabled.False a podmínkou, která časovač zapne nebo nechá vypnutý. Smyčky s podmínkou na konci lze nahradit časovači s TimerEnabled.True s podmínkou, která časovače vypne, nebo ponechá zapnuté. Lze použít i jiné techniky pro náhraducyklů.

Na všechny bloky, které reagují na události, lze také pohlížet jako na smyčky programu s podmínkou na začátku.

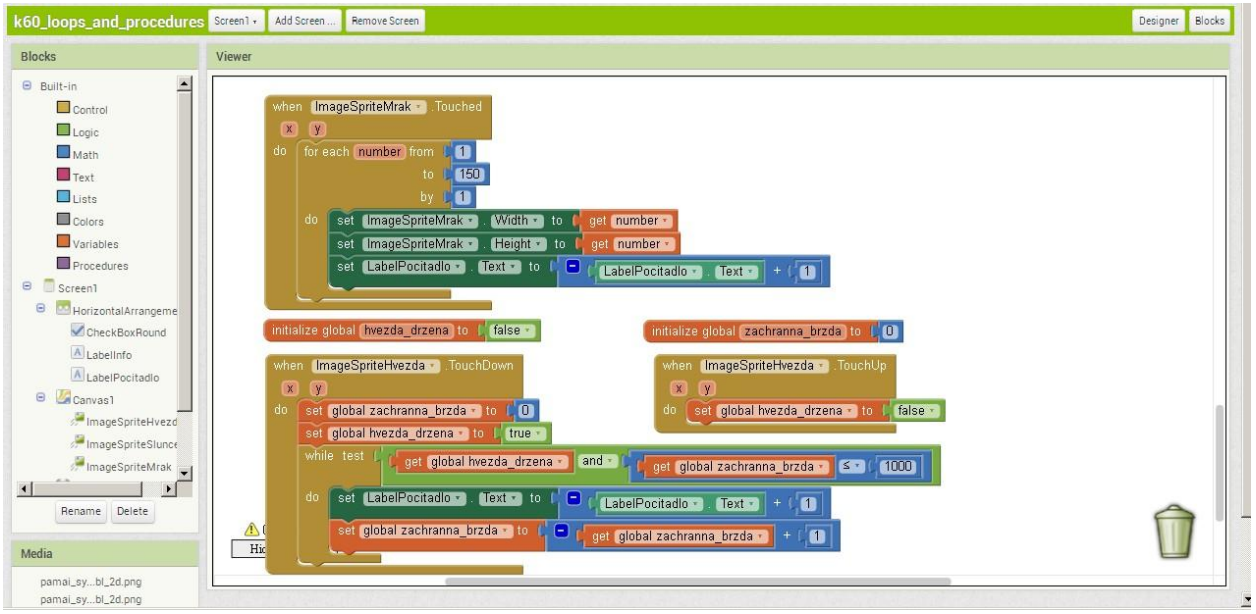
Komponenty modelové aplikace.



Ukázka omezení programů z App Inventoru

Blok WhenImageSpriteMrak.Touched je spuštěn po dotyku obrázku mraku. Jde o počítaný cyklus od 1 do 150 s krokem po 1. Vykoná se tedy 150x. Důležité je si všimnout, že není vidět postupné provádění bloku a zvětšování mraku z velikosti 1 na velikost 150.
Program se (dle výkonu zařízení) na chvíli "zasekne" a pak se objeví už zvětšený mrak. Po celou dobu, kdy byl blok prováděn, nedostal program ani možnost překreslovat sprite mraku a zobrazil až výsledek po skončení bloku.

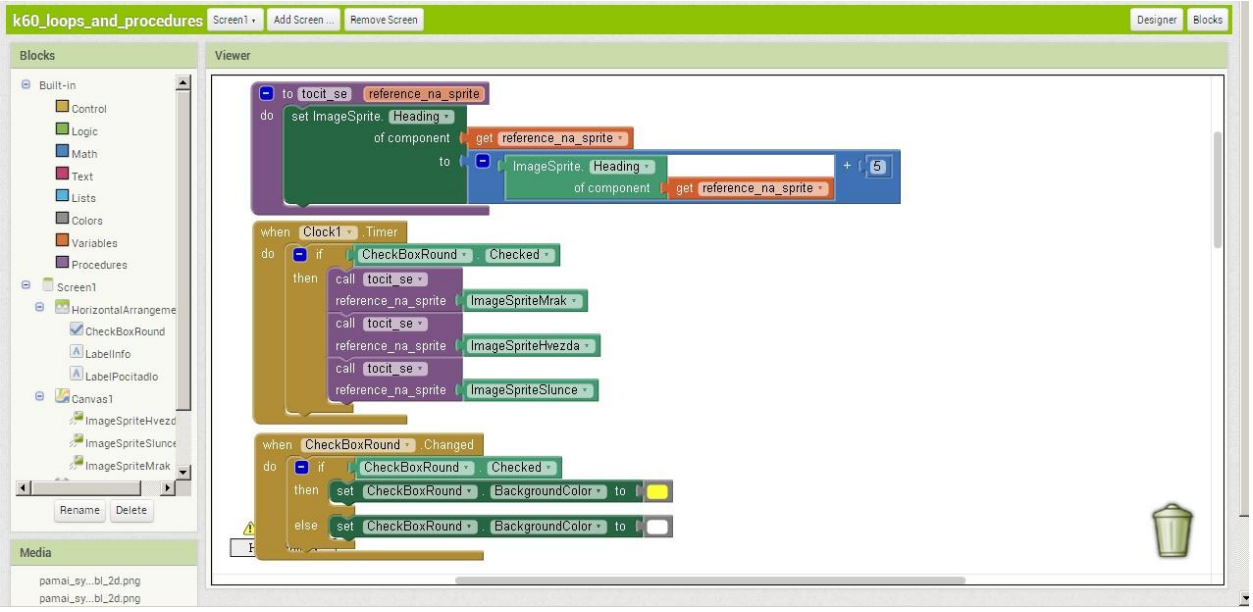
Blok WhenImageSpriteHvezda.TouchDown se samotným SpriteHvezda neprovádí nic, ale po dobu držení hvězdy by měl zvyšovat počítadlo průběhu cyklu a měl by přestat přičítat, když je držení ukončeno. To by měl zajistit blok WhenImageSpriteHvezda.TouchUp. Ve skutečnosti ale program na ukončení dotyku nereaguje a zastaví ho až proměnná zachranna_brzda.



Procedury a princip DRY

Procedury patří k významným řídícím strukturám programu. Umožňují vykonávat stejný kód pro různé objekty. Někdy jsou procedury použity pro oddělení a logické členění zdrojového kódu, což je opět velmi užitečná funkce. Nezřídka o "životnosti" a úspěšnosti programu rozhoduje právě jeho dobrá čitelnost a přehlednost. Na obrázku je ukázka procedury tocit_se, které je jako parametr předána reference na sprite. Funkce zajišťuje otáčení sprite a je jí jedno, jaký sprite je jí předán. Místo tří téměř identických kódů pro 3 sprite v tomto případě, nebo v jiném programu, třeba pro 100 sprite, je použit jeden kód pro všechny.

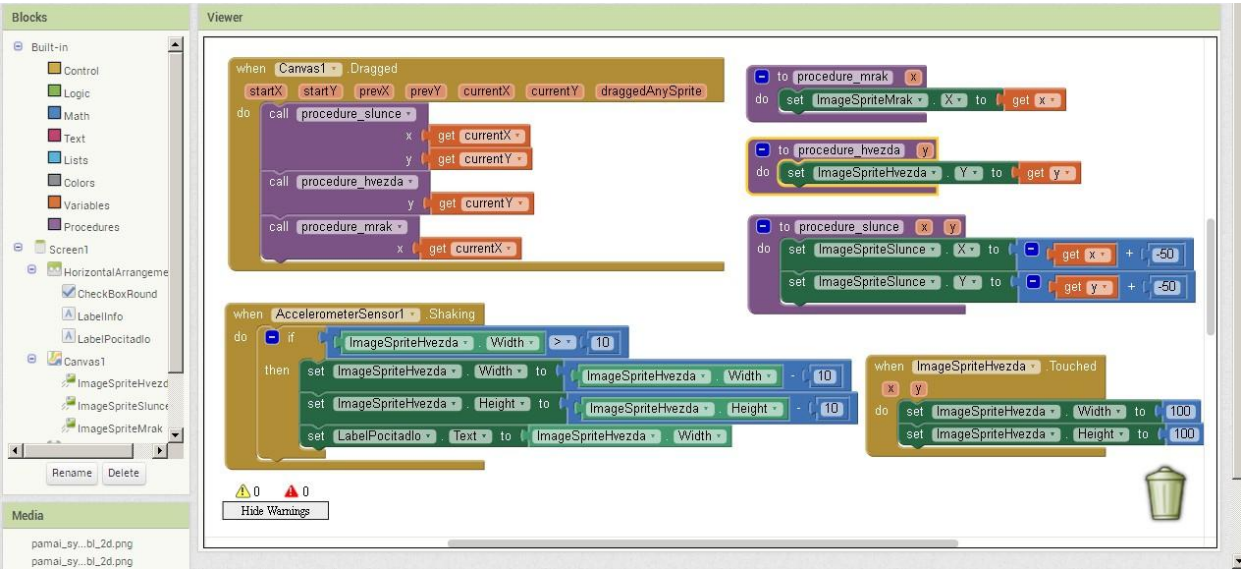
Otáčení je ještě podmíněno zaškrtnutím checkboxu. Jde tedy o variaci na cyklus s podmínkou na začátku.



Ukázka procedur a použití senzoru akcelerátoru

V poslední části programu je obsluha pohybu objektů po ploše opět rešená pomocí funkcí, které zde plní funkci oddělení kódu. Poslední dva bloky pracují se senzorem akcelerátoru. Při třesení zařízením se hvězda zmenšuje, po dotyku se hvězda zvětší.

Funkce AcelerationChanged má v současné verzi (únor 2015 AI nb140b Companion 2.24) chybu, která při použití způsobuje zaseknutí zařízení.



Tipy na modifikace programu:

více sprites s různým chování,
jiná společná akce všech sprites, například hromadné zmenšení při stisku tlačítka, sprite pohybující se po určené dráze,
sprite pohybující se po náhodné dráze, malování sprites na canvas.

10 Aktivita 7 - datové seznamy a komponenty pro práci se seznamy

Tematický celek	Datový typ seznam a komponenty pro práci se seznamy.	
Motivační rámec	Usnadnění práce s větším množstvím proměnných a zjednodušení programu.	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Teoretické a praktické seznámení s datovým typem seznam "list".	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti se seznámí s použitím datového typu seznam a komponentami pro práci se seznamy a vyzkouší si je.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-15m	Seznámení s tématem lekce. Volitelně předvedení a popis ukázkové aplikace. Dle znalostí studentů zopakování informací o datovém typu seznam a základních operací s tímto datovým typem nebo seznámení s nimi.	Frontální výklad, ukázka.
25-50m	Prohlídka ukázkových aplikací. Modifikace ukázkových aplikací případně využití znalosti seznamů při tvorbě vlastní aplikace.	Samostatná a skupinová práce s využitím získaných znalostí, infromací v tomto kurzu a externích zdrojů.
15-30m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Video o vytváření seznamu v AI2 2 min . Creating lists in App Inventor 2 (viz. on-line kurz)

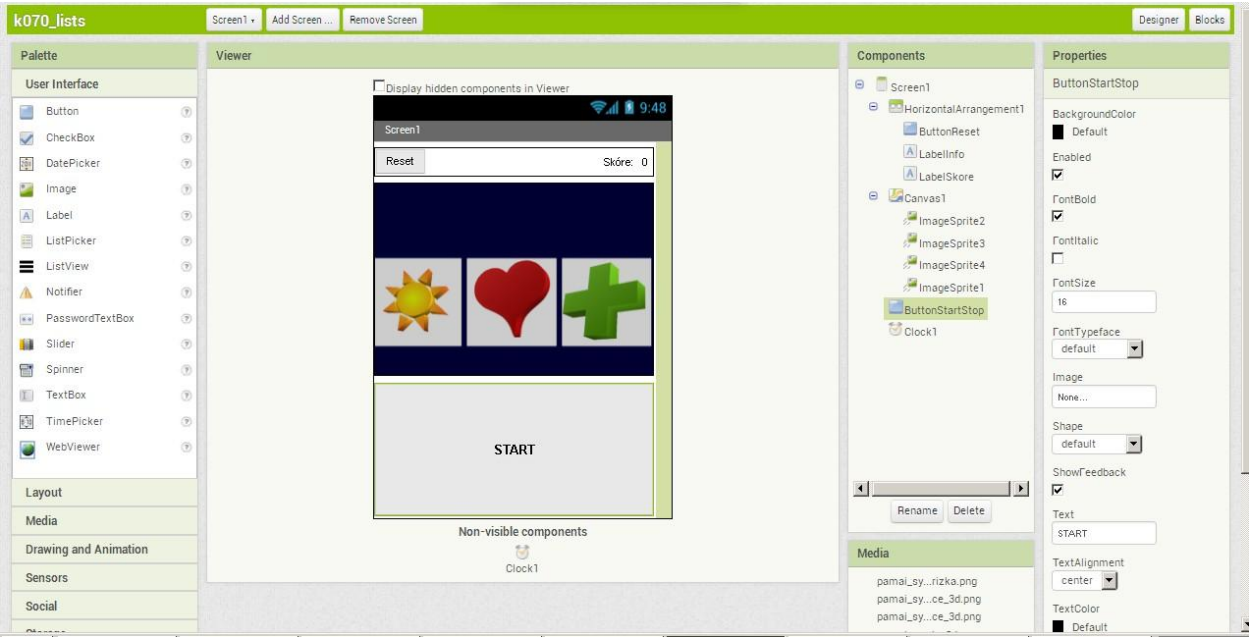
Tutoriál k vytváření seznamu. Making Lists (App Inventor 2) (viz. on-line kurz)

Popis jednotlivých bloků seznamu. AI2 Lists. Aplikace využívající seznamy. Math Blaster (viz. on-line kurz)

Aplikace pro ukázku práce s datovými seznamy

Datové seznamy usnadňují práci s programem. Podobně jako procedury zobecňují kód a umožňují ho zjednodušit, seznamy zjednodušují práci s proměnnými a jejich obsahem. Aplikace napodobuje hru videostop. Po stisknutí tlačítka start se mění obrázky tří sprite a talčítkem stop ze změny zastaví. Hráč zvedá skóre, když se mu podaří zastavit na stejných obrázcích.

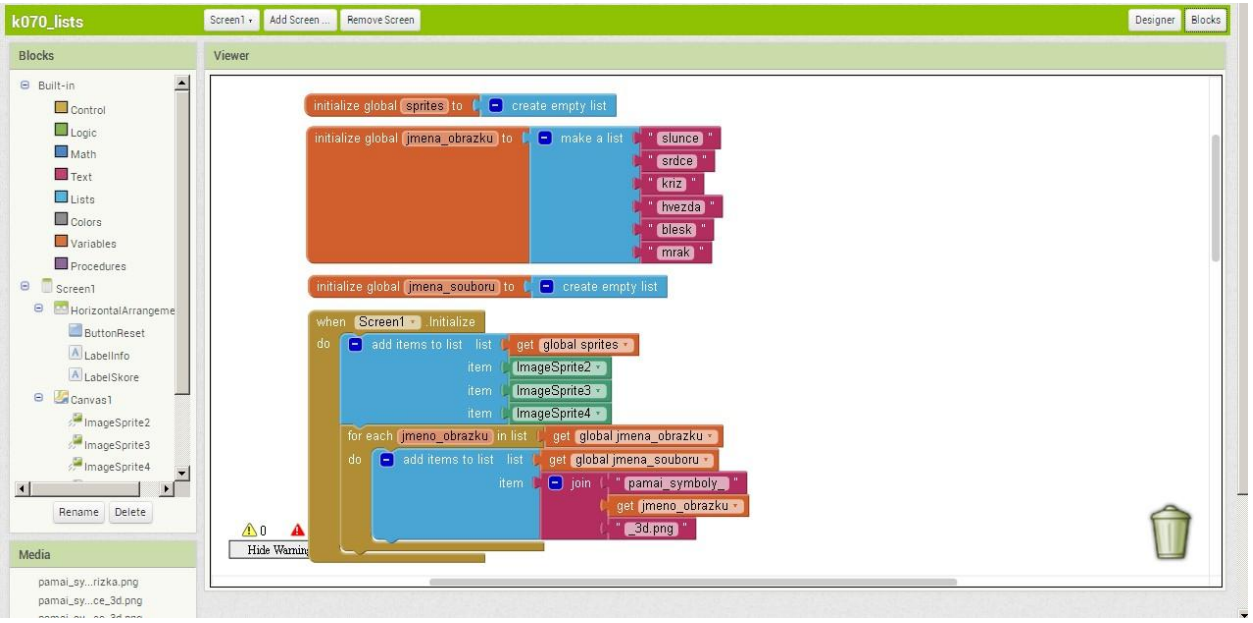
Komponenty modelové aplikace



Inicializace seznamů

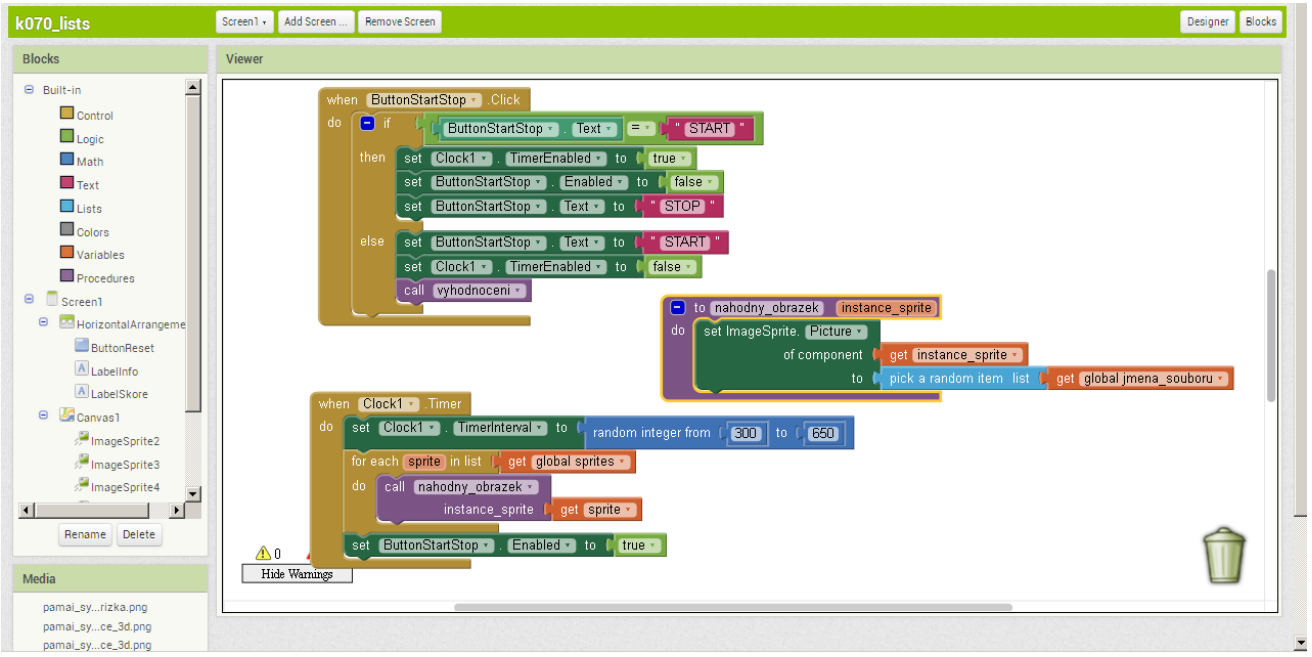
V ukázce jsou vytvořeny tři seznamy. sprites a jmena_souboru jsou inicializovany jako prázdné seznamy a jsou naplněny v bloku when Screen.Initialize.

Seznam jmena_obrazku je naplněn při inicializaci. Obsahuje řetězce, kterými se jednotlivé obrázky liší. V bloku when Screen.Initialize je nejdříve vytvořen seznam instancí sprites, následně je vytvořen seznam jmena_souboru, který obsahuje celé názvy souborů s obrázky. Seznamy tedy mohou obsahovat nejen čísla a textové řetězce, mohou obsahovat také instance objektů.



Cyklus nad objekty seznamu

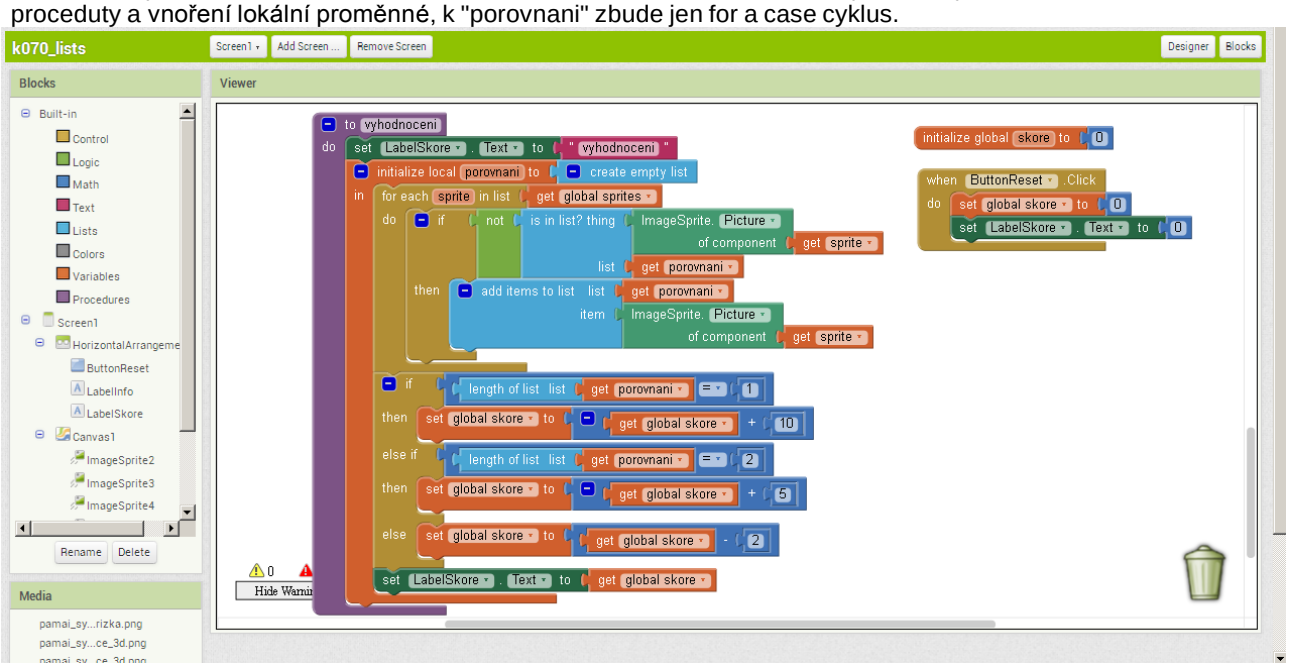
Tlačítkem ButtonStartStop je řízeno pouštění a zastavení časovače. Časovač Clock1 spouští v cyklu "for each sprite in list get global sprites" cyklus nad dříve vytvořeným seznamem sprites. Pro každý sprite zavolá proceduru nahodny_obrazek, která vybere ze seznamu obrázků jeden náhodný. Po druhém stisknutí tlačítka ButtonStartStop je zavolána funkce pro vyhodnocení skóre.



Vyhodnocení výsledku

Uvnitř procedury vyhodnoceni je vytvořen seznam jako lokální proměnná. Pak potřebujeme zjistit, kolik obrázků je stejných. Místo porovnávání obrázků každého s každým je tento úkol řešen pomocí seznamu. V cyklu for each se projdou všechny zobrazené obrázky a přidávají se do seznamu, pokud tam ještě nejsou. Pokud jsou po cyklu for each v seznamu 3 obrázky, znamená to, že hráč nestrefil ani dva stejné. Pokud jsou v seznamu dva obrázky, hráč strefil dva stejné a jeden jiný obrázek. Pokud je v seznamu jen jeden obrázek, znamená to, že hráč strefil 3 stejné obrázky.

Procedura vyhodnocení může vypadat na první pohled komplikovaně, ale když si odmyslíte dvě vrstvy, vnoření samotné proceduty a vnoření lokální proměnné, k "porovnani" zbude jen for a case cyklus.



Tipy na modifikace programu:

více než 6 obrázků, které se střídají,
více než 3 sprites pro zobrazování obrázků,
náhodné zastavení obrázku a ponechání beze změny,
počítání počtu pokusů, případně omezení na x pokusů, pak reset.

11 Aktivita 8 - aplikace s více obrazovkami a správci rozložení

Tematický celek	Tvroba aplikace s více obrazovkami a kombinovaní správců rozložení.	
Motivační rámec	Získání větší kontroly nad vzhledem a organizací aplikace.	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Prohloubení znalostí o chování a využití správců rozložení "layout managerů". Teoretické a praktické seznámení s konceptem více obrazovek "screens" v rámci jedné aplikace.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti si vyzkouší kombinaci layout managerů. Studenti se seznámí s konceptem obrazovek a vyzkouší si vyrobit vlastní aplikaci s více obrazovkami.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-25m	Seznámení s tématem lekce. Volitelně předvedení a popis ukázkové aplikace. Popis jednotlivých správců rozložení, jejich kombinování a ovlivnění jejich chování pomocí parametrů, zarovnání, výška a šířka. Seznámení s konceptem obrazovek. Seznámení s omezeními při sdílení kódu a proměnných mezi obrazovkami.	Frontální výklad, ukázka.
25-50m	Prohlídka ukázkových aplikací. Modifikace ukázkových aplikací, případně využití znalosti seznamů při tvorbě vlastní aplikace.	Samostatná a skupinová práce s využitím získaných znalostí, infromací v tomto kurzu a externích zdrojů.
15-30m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Informace ke správcům rozložení. Informace k blokům kódu řídících screen. (viz. on-line kurz)

Video-tutoriál k tvorbě více obrazovek v jedné aplikaci. (viz. on-line kurz)

Video-tutoriál k tvorbě více "virtuálních" obrazovek v jedné aplikaci pomocí skrývání bloků správce rozložení. Creating Multiple Screens with App Inventor (viz. on-line kurz)

Informace ke správcům rozložení a obrazovkám

Správci rozložení "layout managers" pomáhají uspořádat objekty grafického prostředí "widgety". AI2 nabízí 3 správce rozložení, verikálního, horizontálního a tabulkového. Tito správci mohou být zanořeni do sebe. Důležité pro určení chování widgetů jsou i další parametry, jako je určení šířky / výšky, případně vyplnění celého prostoru nadřazené komponenty, vertikální a horizontální zarovnání.

Obrazovky "Screens" jsou významný koncept práce s aplikací, v prostředí Androidu velmi rozšířený. Na screen lze v AI2 pohlížet jako na samostatnou aplikaci. V editoru má svůj vlastní design a má svůj vlastní kód. S jinými obrazovkami ve stejné aplikaci nesdílí ani proměnné. Na práci s obrazovkami je třeba si v AI2 trochu zvyknout. Při práci s návrhářem se vám v zařízení nebo emulátoru přepínají obrazovky podle toho, který máte v AI2 vybraný. Ne vždy je reakce na přepnutí rychlá, nezřídka k ní vůbec nedojde. Bohudík výrazně lepší odezva a funkčnost je při použití vyexportované .apk.

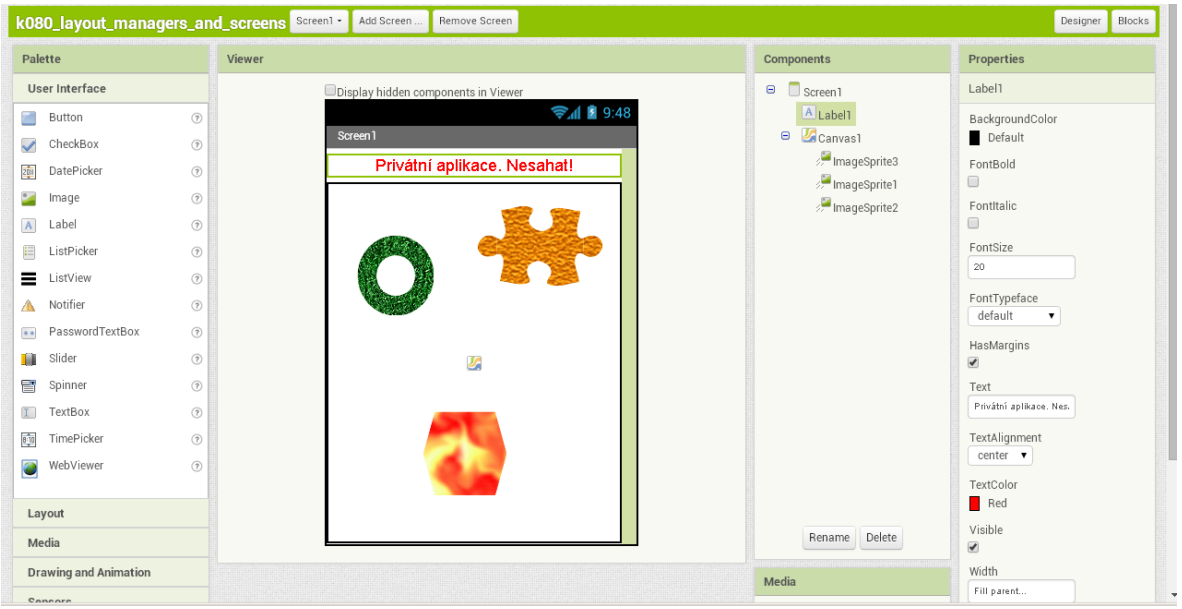
Názvy obrazovek nemohou být přejmenovány, proto se vyplatí vhodnému názvu věnovat patřičnou pozornost. Screen1 nemůže být smazán.

Aplikace pro ukázku práce s vnořenými správci rozložení a s více obrazovkami

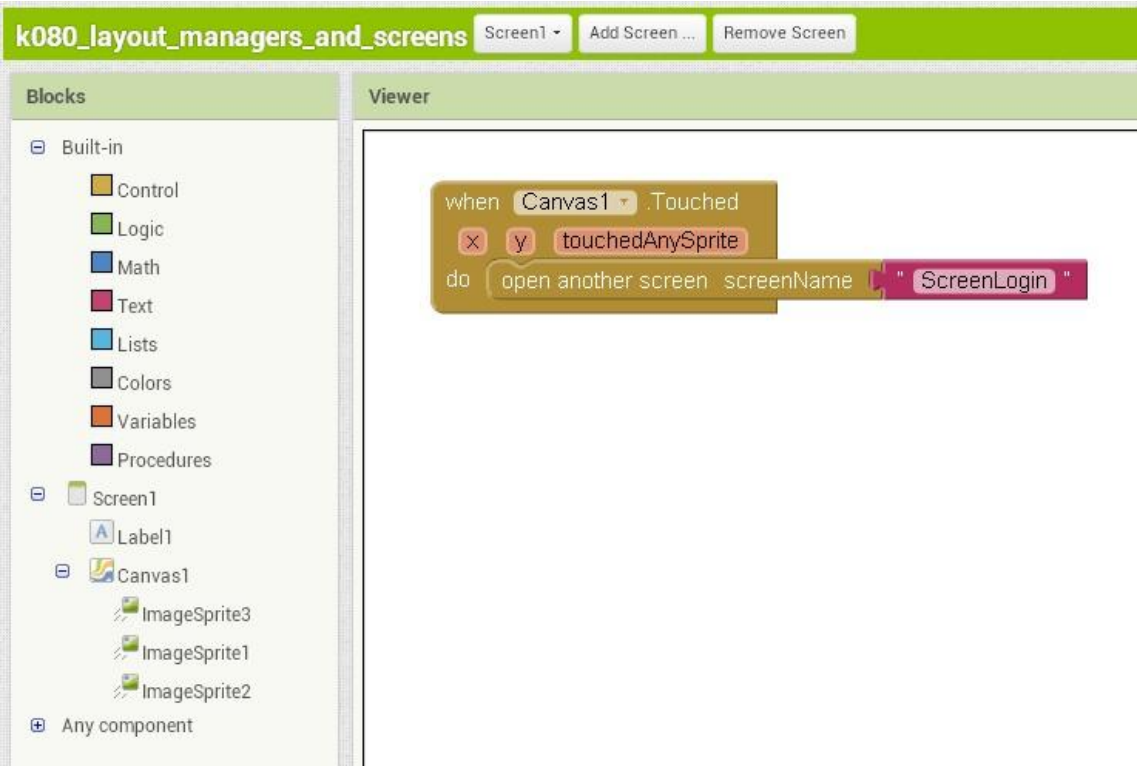
Práce se správci rozložení a s obrazovkami je ukázána na aplikaci, která zabezpečuje přístup k "tajnému" obsahu. Aplikace generuje klávesnici s náhodným zamícháním kláves, aby nebylo snadné vysledovat heslo. Aplikace umožňuje zadat i heslo pro případ nebezpečí. Heslo je přijato, ale místo tajného obsahu je zobrazen falešný obsah.

Úvodní obrazovka modelové aplikace

Tato obrazovka po dotyku plochy zavolá obrazovku s přihlašovacím dialogem.

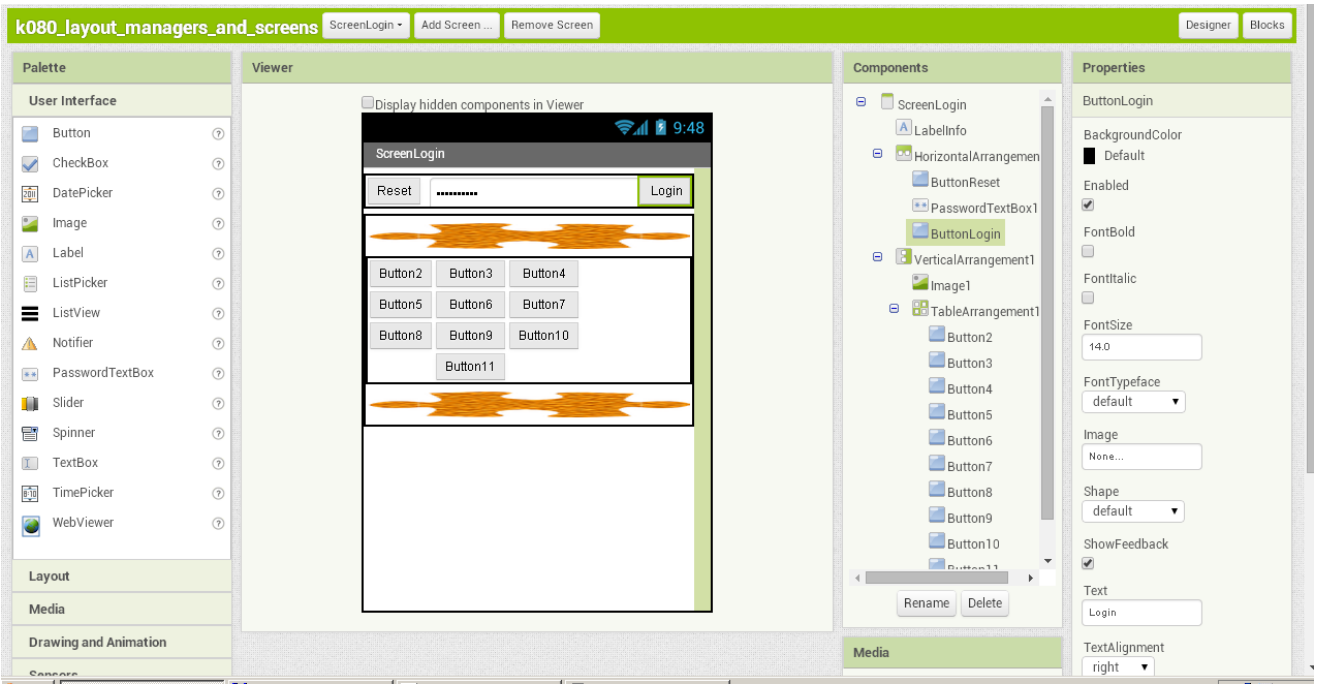


Kód obsahuje jediný blok, který zavolá obrazovku "ScreenLogin".



Obrazovka pro přihlášení uživatele

Je zde ukázáno použití vnoření vertikálního a tabulkového správce rozložení. Poprvé je použito také políčko pro zadání hesla. Chová se stejně jako prosté textové pole, ale místo znaků zobrazuje tečky.

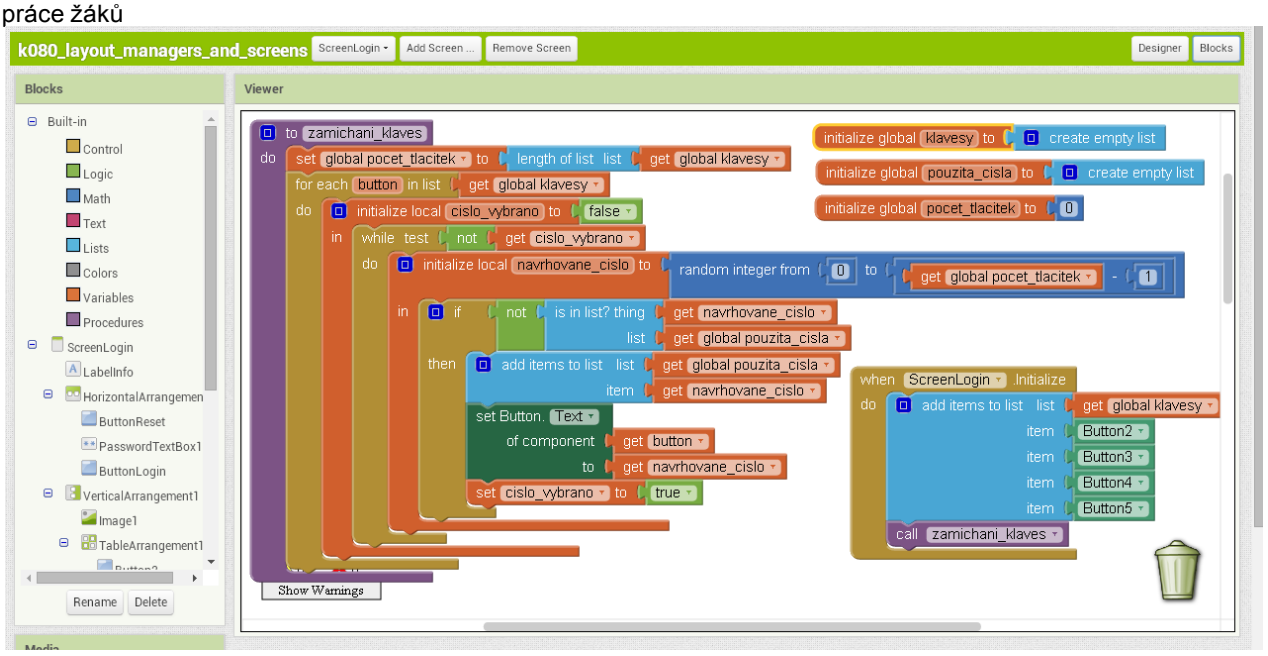


Kód obrazovky pro přihlášení

Tomuto kódu dominuje procedura pro zamíchání číslic na klávesách. Postup je následující: Nejdříve jsou vytvořeny 2 seznamy. Seznam klavesy nelze naplnit ihned a je třeba ho plnit až v bloku inicializace obrazovky. Seznam je naplněn instancemi tlačítek a pak je zavoláno zamíchání.

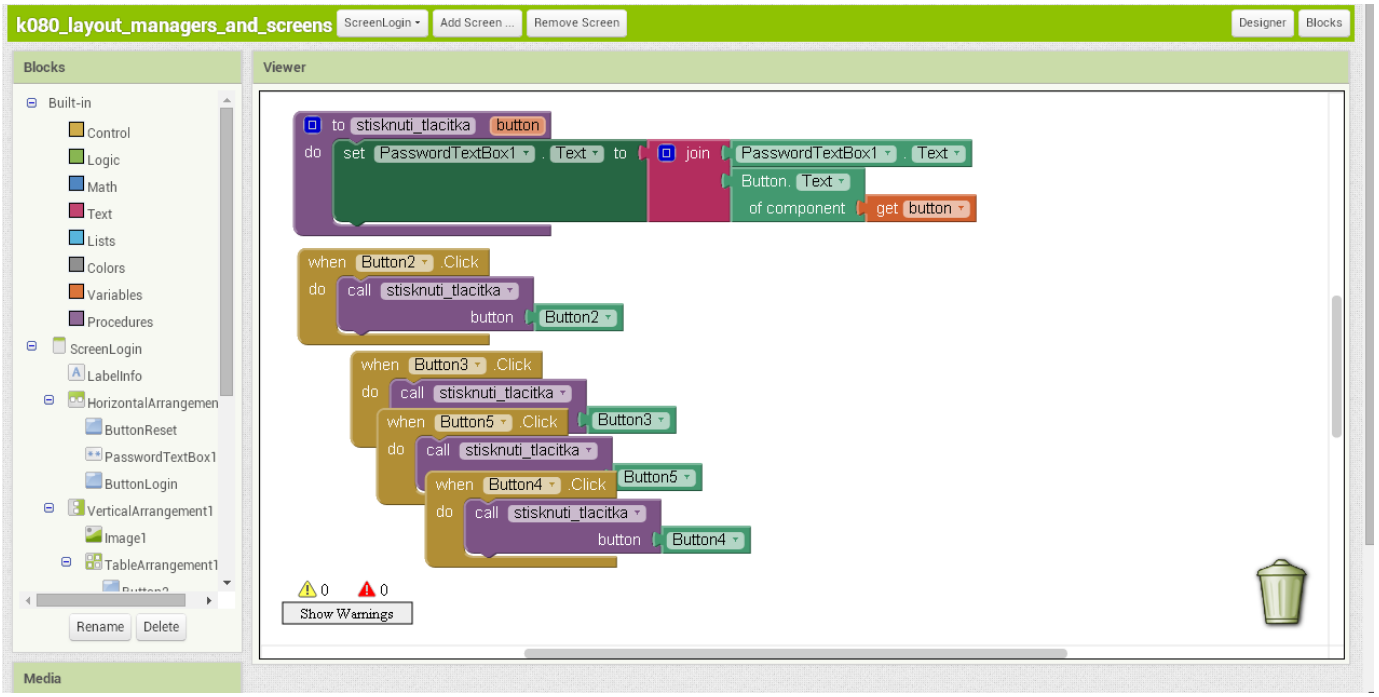
Pro každé tlačítko / button je volán cyklus, který vybere náhodně jednu číslici, kterou tlačítku přiřadí. Přiřazenou číslici dále nechceme přiřazovat dalšímu tlačítku, a tak je vytvořen seznam pouzita_cisla, do kterého se použitá čísla ukládají. Cykus while test se opakuje tak dlouho, dokud není pomocí random vybráno takové číslo, které ještě žádné tlačítko nemá.

V programu se náhodná čísla generují jen pro 4 tlačítka. Obsluha dalších tlačítek může být předmětem samostatné práce žáků



Zápis vybraných znaků do pole password

Uživatel může zapisovat heslo rovnou do pole password, ale pokud píše na zabezpečené klávesnici, pak je třeba obloužit předávání stisknutých kláves. Pro všechna tlačítka jde o stejnou akci, kterou obsluhuje procedura stisknuti_tlacitka. Pokud je v AI2 mnoho stejných bloků, doporučuji si je poskládat přes sebe a ušetřit místo.

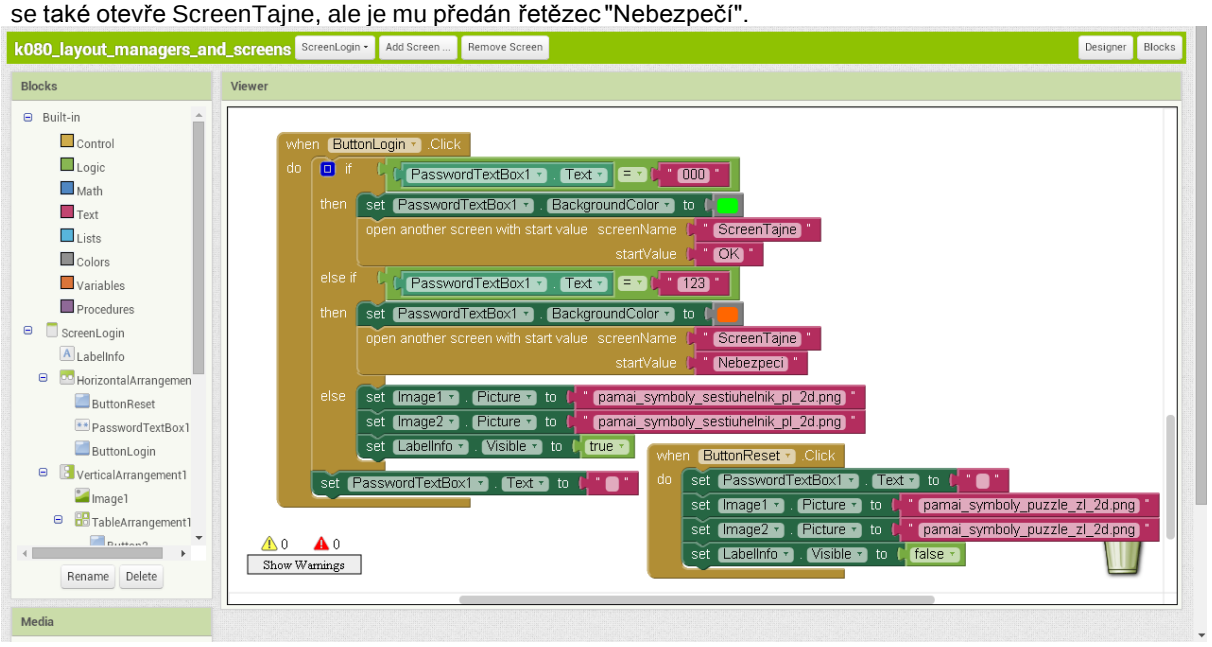


Ověření hesla

V poslední ukázce kódu se řeší ověření hesla a reset hodnot.

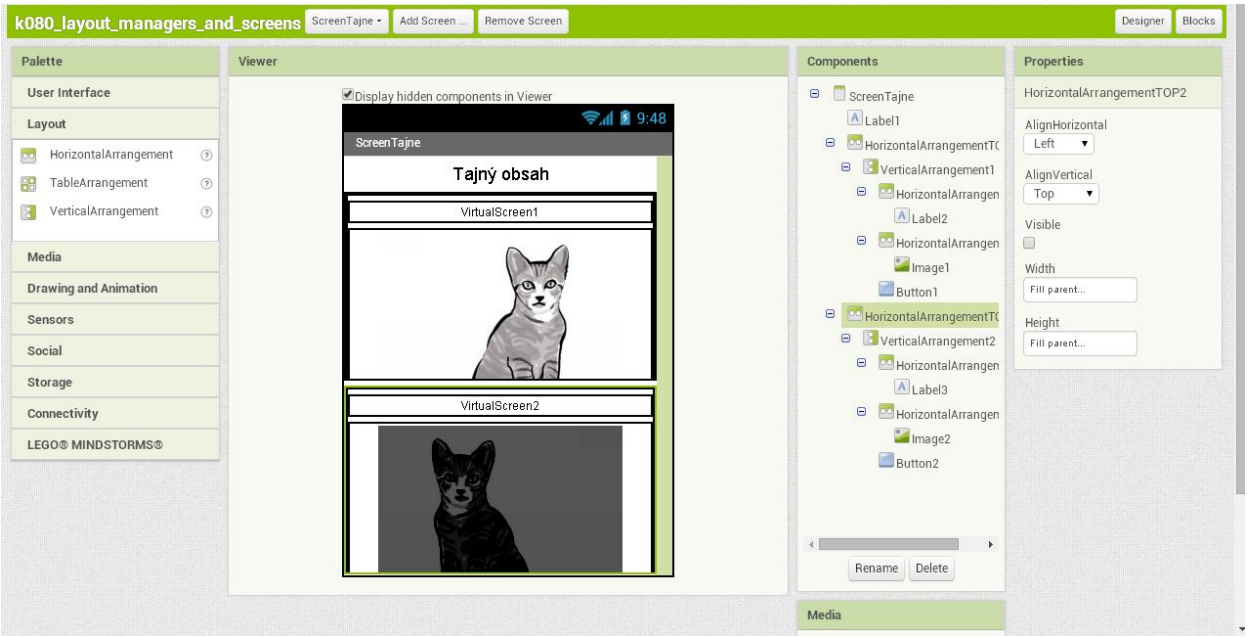
Předem je třeba přiznat, že zabezpečení hesla je u této aplikace velmi slabé. Takto zadané heslo by případný útočník, který by se dostal k .apk naší aplikace, velmi snadno našel ve zdrojovém kódu. V praxi se heslo neporovnává přímo s konkrétním řetězcem, ale bývá uložena jeho zašifrovaná podoba.

V této ukázce nám však jde o managery rozložení a o práci s obrazovkami. Při zadání správného hesla "000" (můžete samozřejmě libovolně změnit) se otevře obrazovka s tajným obsahem. Při zadání bezpečnostního hesla se také otevře ScreenTajne, ale je mu předán řetězec "Nebezpečí".



Obrazovka s tajným obsahem

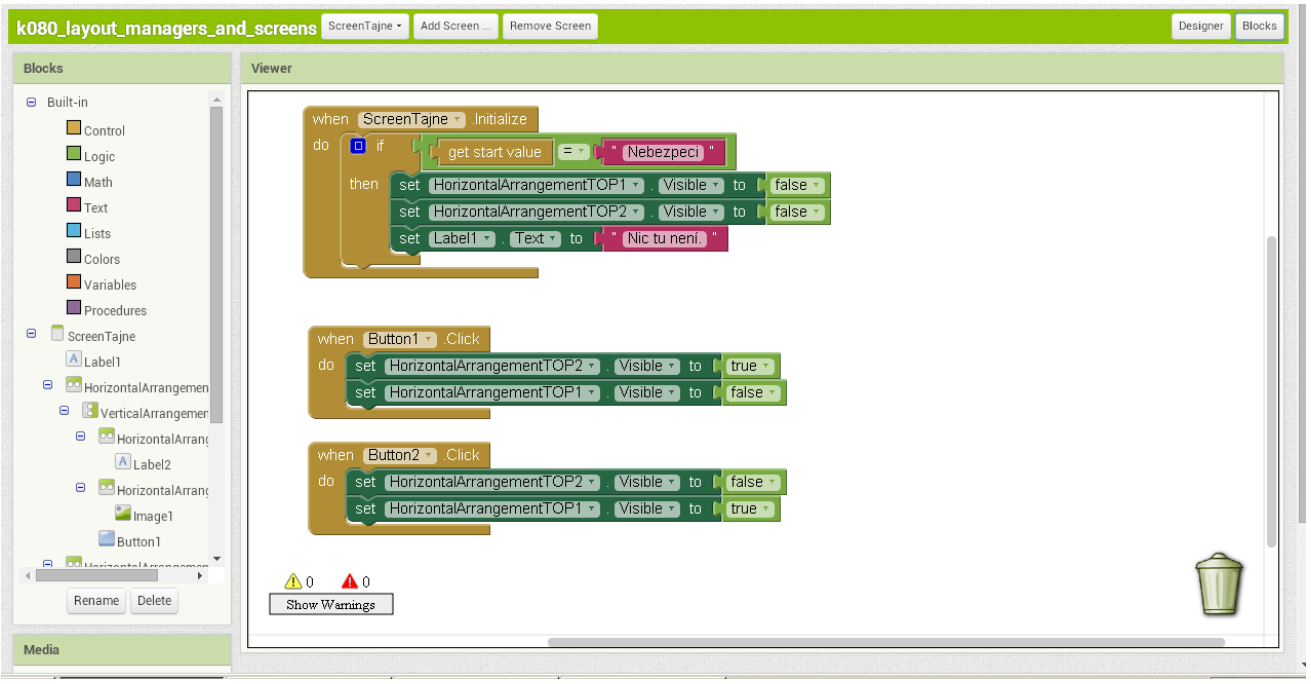
Obrazovka s tajným obsahem využívá ještě více správce rozložení. Používá je i k akci tzv. "virtuálních obrazovek", kdy pouze jeden vrcholový layout manager je zobrazen, zatímco ostatní jsou skryti. Tím lze vytvořit iluzi více obrazovek. Na obrázku si všiměte, že "Display hidden component in Viewer" nad ukázkou obrazovky je zapnutý a v properties je pro HorizontalArrangementTOP2.visible vypnuto.



Kód obrazovky s tajným obsahem

V bloku when ScreenTajne.initialize se vyhodnocuje, zda byla obrazovka otevřena bezpečnostním heslem. Pokud ano a obdržela žetězec "Nebezpecí", pak zajistí skrytí tajného obsahu a přidá falešný obsah "Nic tu není."

Následující dva bloky se starají o přepínání mezi virtálními obrazovkami.



Tipy na modifikace programu:

doplnění funkce bezpečnostní klávesnice pro další tlačítka,
vyladění vzhledu bezpečnostní klávesnice, např. zarovnání na střed obrazovky, velikost a rozmístění tlačítek, přidání vlastní obrazovky, na kterou by byl uživatel odkázán při zadání jiného hesla,
drobné zabezpeční hesla, aby nebylo v programu jako viditelný text "plain text".

12 Aktivita 9 - ukládání dat na paměťové kartě "storage"

Tematický celek	Použití komponent pro ukládání dat na kartě / disku, a jejich načtení.	
Motivační rámec	Seznámení s důležitým aspektem programování, ukládání a načítáním dat do non-volatilní paměťi.	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Praktické vyzkoušení ukládání dat a jejich načtení v prostředí App Inventor.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti se seznámí a vyzkouší si použití cyklů programu.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-25m	Seznámení s tématem lekce. Volitelně teorie o ukládání dat v PC (složky, soubory, databáze). Volitelně ukázka modelové aplikace a popis jednotlivých bloků, týkajících se práce s daty. Volitelně samostatná činnost žáků, vyhledávání informací k tomuto	Frontální výklad, ukázka, samostatná práce žáků
30-60m	Analýza, úpravy a modifikace modelové aplikace, volitelně zakomponování funkcí pro ukládání a načítání dat do vlastní aplikace.	Samostatná a skupinová práce s využitím získaných znalostí, infromací v tomto kurzu a externích zdrojů.
15-20m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postup a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Wikipedie k tématu ukládání dat. (viz. on-line kurz)

Doplňkový tutoriál s aplikací malování. Práce s canvas, sprites, více obrazovek, vlastní procedury, ukládání dat.

(viz. on-line kurz)

Popis komponent storage. (viz. on-line kurz)

Aplikace využívající ukládání a načítání dat z paměťové karty

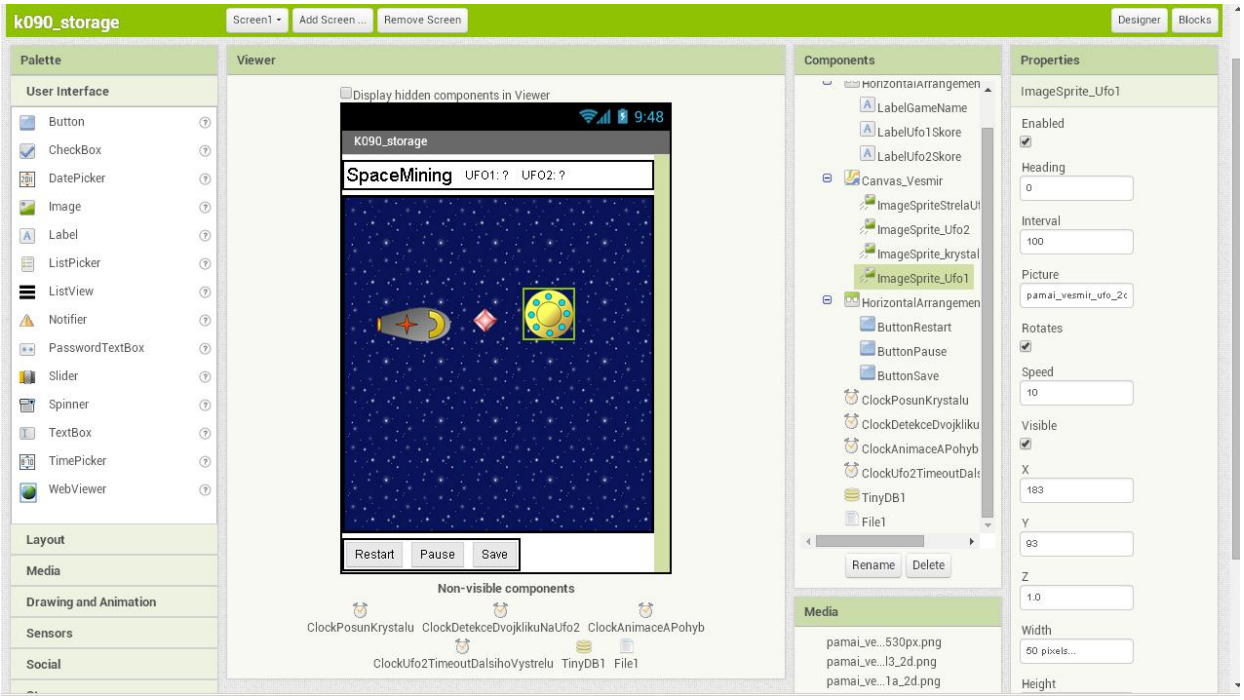
AI2 používá k ukládání a načítání dat několik komponent. V ukázkové aplikaci budeme používat dvě komponenty. TinyDB a File. TinyDB je databáze vázaná ke konkrétní aplikaci. Protože všechny aplikace spouštěné z AI2 pomocí AI Companion jsou z pohledu systému jedna aplikace, je třeba pro ukládání do TinyDB dostatečně unikátní slova, aby si aplikace navzájem nepřepisovaly data. Storage file umožňuje ukládat data do souborů přímo do souborového systému.

Rozšíříme funkce aplikace z aktivity 6 o ukládání skóre hráčů a uložení snímku aplikace. Většina kódu aplikace zůstala stejná, bloky, které byly změněny nebo doplněny, jsou v této ukázce.

V minulé aktivitě bylo zmíněno, že jednotlivé screeny nesdílí proměnné. Sdílení proměnných mezi screeny se dá vyřešit právě pomocí TinyDB.

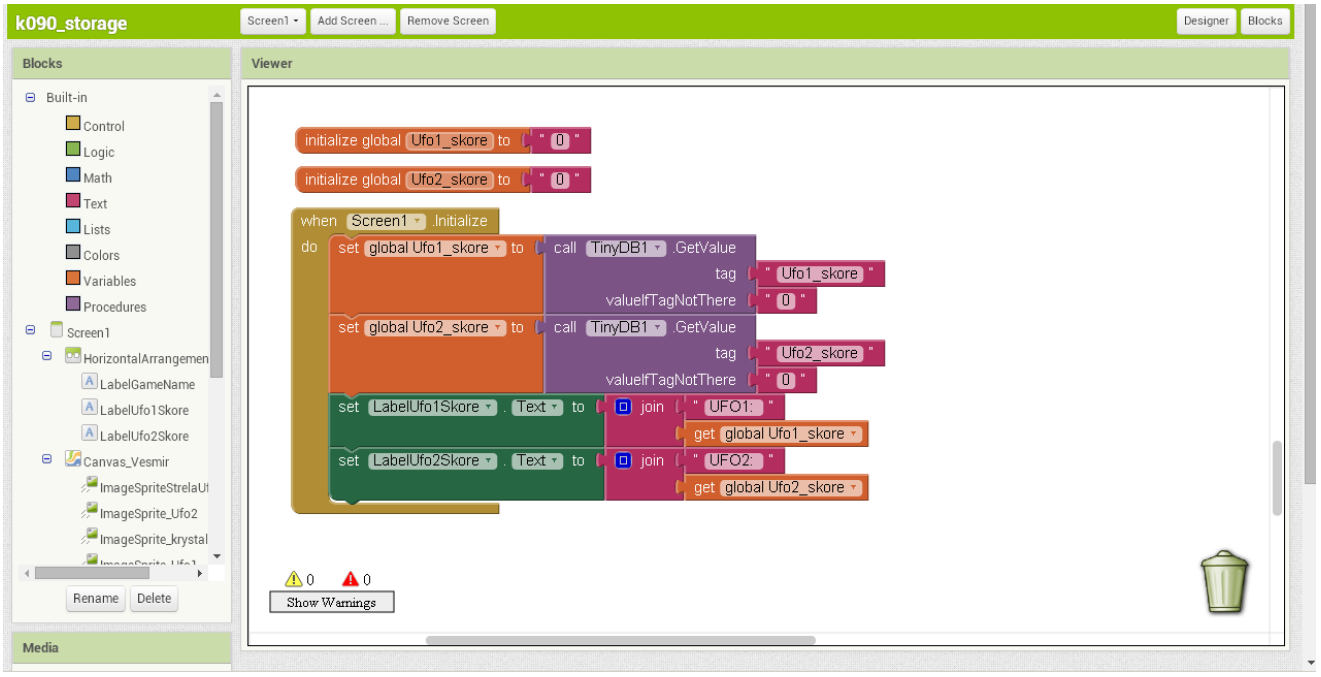
Komponenty modelové aplikace

U komponent se mnoho nezměnilo. Tlačítko Restart bylo přesunuto vlevo dolů a přibyla tlačítka Pause a Save. Významnou změnou jsou však komponenty TindyDB1 a File1.



Inicializace a načtení dat z databáze

Protože nelze proměnné inicializovat přímo hodnotami z databáze, je konstrukce následující. Nejdříve jsou vytvořeny globální proměnné. Následně, v bloku when Screen1.Initialize jsou načteny a předány hodnoty z databáze. Pokud by v databázi požadavaná data nebyla, použije se výchozí hodnota z valueTagNotThere. Na závěr inicializace jsou hodnoty zapsány do LabelUfo1Skore a LabelUfo2Skore.

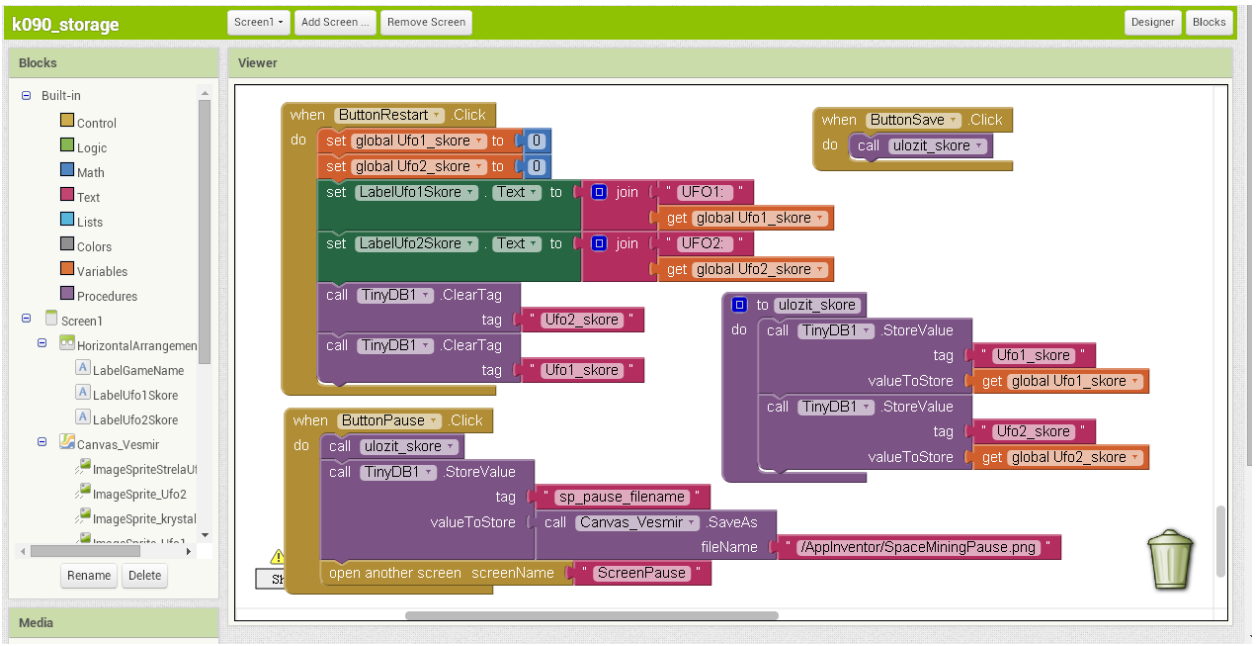


Bloky využívající ukládání dat a mazání dat

Blok when ButtonRestart.click řeší vynulování obsahu globálních proměnných, zobrazení dat v LabelUfo1Skore a LabelUfo2Skore a stará se i o smazání dat v databázi.

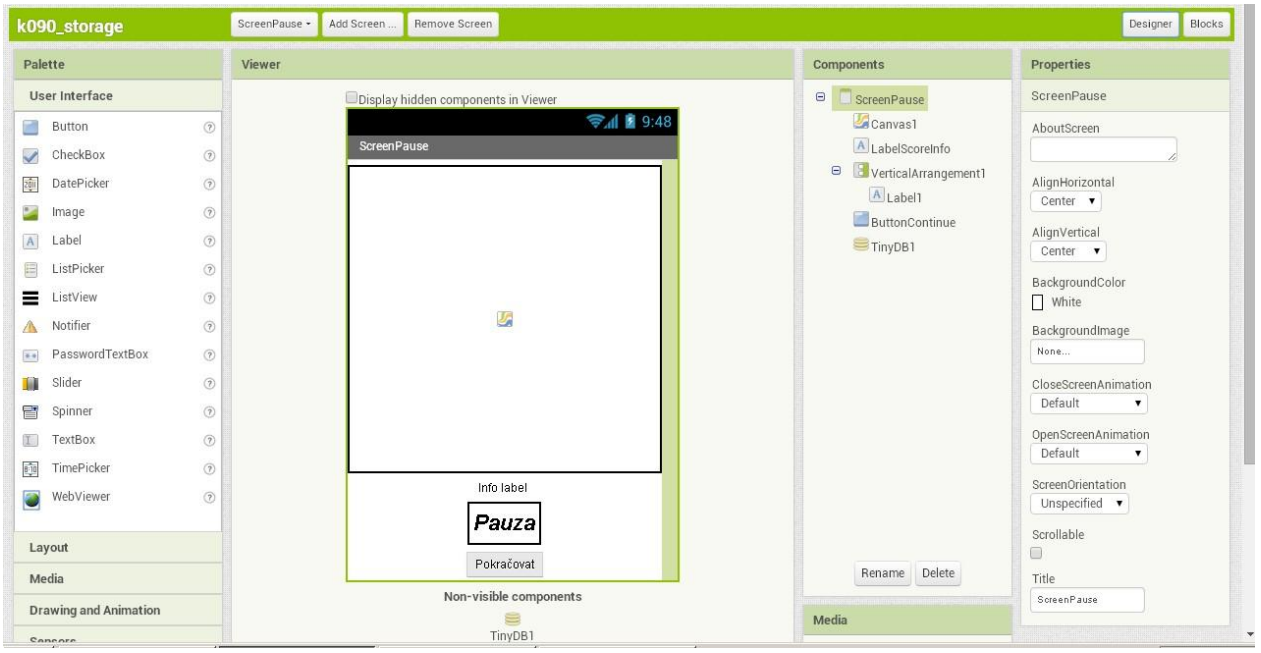
When ButtonSave.Click volá proceduru ulozit_skore, která zapíše hodnoty globálních proměnných do databáze. V databázi jsou hodnoty uloženy pod názvem (tagem) Ufo1_skore a Ufo2_skore.

V obsluze tlačítka Pause se kromě uložení skóre řeší vyrobení snímku obrazovky, jeho uložení na disk do souboru
/AppInventor/SpaceMining.png uložení cesty a názvu do databáze a otevření obrazovky ScreenPause. Cesta



Design obrazovky ScreenPause

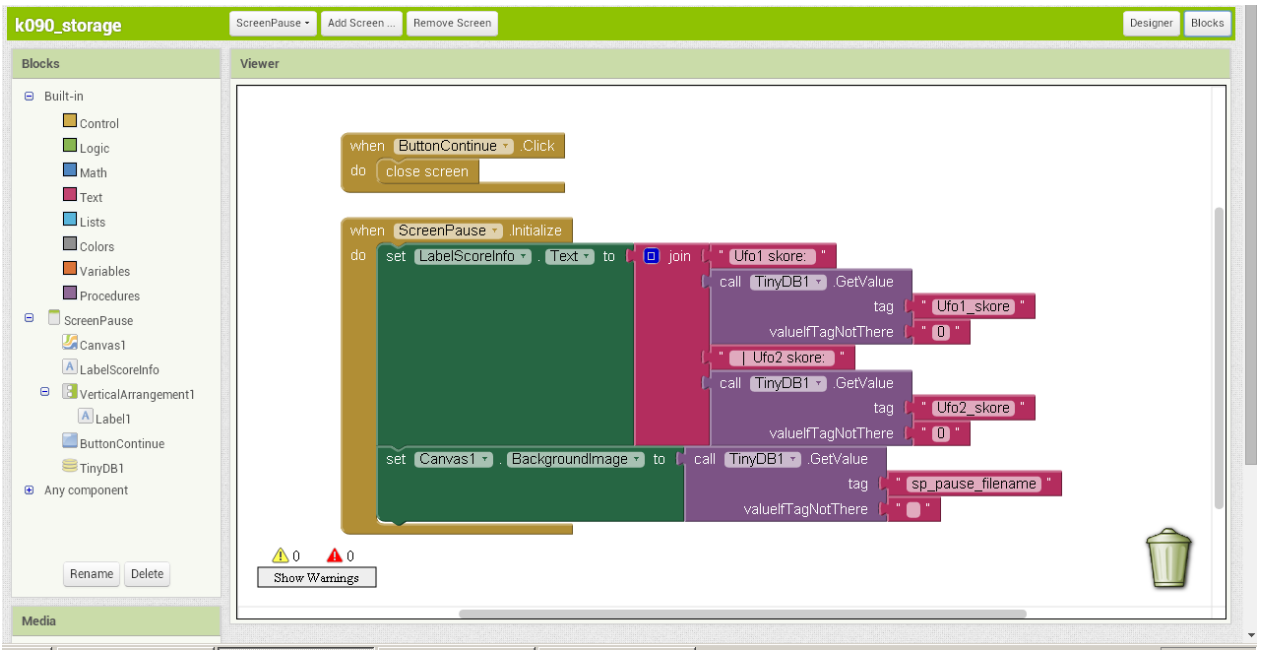
Většinu plochy zabírá komponenta Canvas1, ve které se budou zobrazovat screenshoty hry, před jejím pozastavením. Za povšimnutí stojí opětovné zařazení komponenty TinyDB. Pro každý screen, ve kterém se bude pracovat s uloženými daty v databázi, je třeba znovu přidat komponentu TinyDB.



Bloky obrazovky ScreenPause

Obsluha tlačítka Pokračovat nepoužívá otevření obrazovky Screen1, ale zavírá sama sebe. Kdyby tomu tak nebylo, vznikal by cyklus, který by znesnadňoval použití systémového tlačítka Zpět.

Blok when ScreenPause.Initialize řeší zobrazení skóre hráčů, načtení a zobrazení screenshotu z předchozí obrazovky.



Tipy na modifikace programu:

záznam více údajů, (např. životů, počet srážek, souřadnice lodi a její směr), uložení, načtení a zobrazení těchto údajů, ukládání snímků pomocí dalšího tlačítka,
na obrazovce ScreenPause prohlížení uložených screenshotů, přihlašování ke hře, zapamatování jmen hráčů a jejich skóre, zobrazování jmen hráčů, místo názvů UFO1 UFO2,
výběr lodí, jejichž parametry (obrázek, rychlost pohybu, ...) by byly načítány z databáze.

13 Aktivita 10 - další prvky pro interakci s uživatelem a vlastní webový prohlížeč

Tematický celek	Seznámení s dalšími prvky pro interakci s uživatelem a tvorba vlastního webového prohlížeče.	
Motivační rámec	Rozšíření znlalostí o funkci a použití užitečných prvků grafického prostředí a tvorba vlastního webového prohlížeče.	
Počet žáků	5-15	
Věk žáků	10+	
Pomůcky	PC odpovídajících parametrů, webový prohlížeč Mozilla Firefox nebo Google Chrome, zařízení s Androidem připojené k Internetu, případně emulátor zařízení s Androidem, účet pro Google služby.	
Stručný popis aktivity	Přihlášení k prostředí App Inventoru http://appinventor.mit.edu/ Seznámení s dalšími prvky uživatelského prostředí. Obohacení vlastní aplikace o některé z prvků GUI. Tvorba vlastního webového prohlížeče.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Studenti se seznámí a vyzkouší si použití nových prvků grafického uživatelského prostředí. Studenti si vyzkouší vytvořit vlastní webový prohlížeč.	
Předchozí znalosti	Aktivita navazuje na předchozí lekci tohoto kurzu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
5-25m	Seznámení s tématem lekce. Volitelně teorie o ukládání dat v PC (složky, soubory, databáze). Volitelně ukázka modelové aplikace a popis jednotlivých bloků, týkajících se práce s daty. Volitelně samostatná činnost žáků, vyhledávání informací k tomuto tématu.	Frontální výklad, ukázka, samostatná práce žáků
30-60m	Analýza, úpravy a modifikace modelové aplikace, volitelně zakomponování funkcí pro ukládání a načítání dat do vlastní aplikace.	Samostatná a skupinová práce s využitím získaných znalostí, infromací v tomto kurzu a externích zdrojů.
15-20m	Prezentace prací žáků, sdílení poznatků, diskuze. Motivace žáků k domácí práci.	Prezentace prací a diskuze
Hodnocení	Zpětná vazba je okamžitá. Student reflektuje vzdělávací cíle, svůj postupu a dosažené výsledky. Student průběžně srovnává vlastní výsledky s výsledky ostatních studentů.	
Návaznosti	Na tuto aktivitu navazují následující aktivity, které prohlubují a kombinují znalosti prostředí App Inventoru.	
Poznámky	Pro případ výpadku připojení k internetu nebo technických problémů s App Inventorem doporučuji připravit záložní aktivitu, založenou na podobných principech blokového programování, například offline aplikaci MIT Scratch, BYOB nebo některou z variant programovacích prostředí LOGO.	

Wikipedie k tématu grafického uživatelského prostředí GUI. Popis komponent uživatelských ovládacích prvků

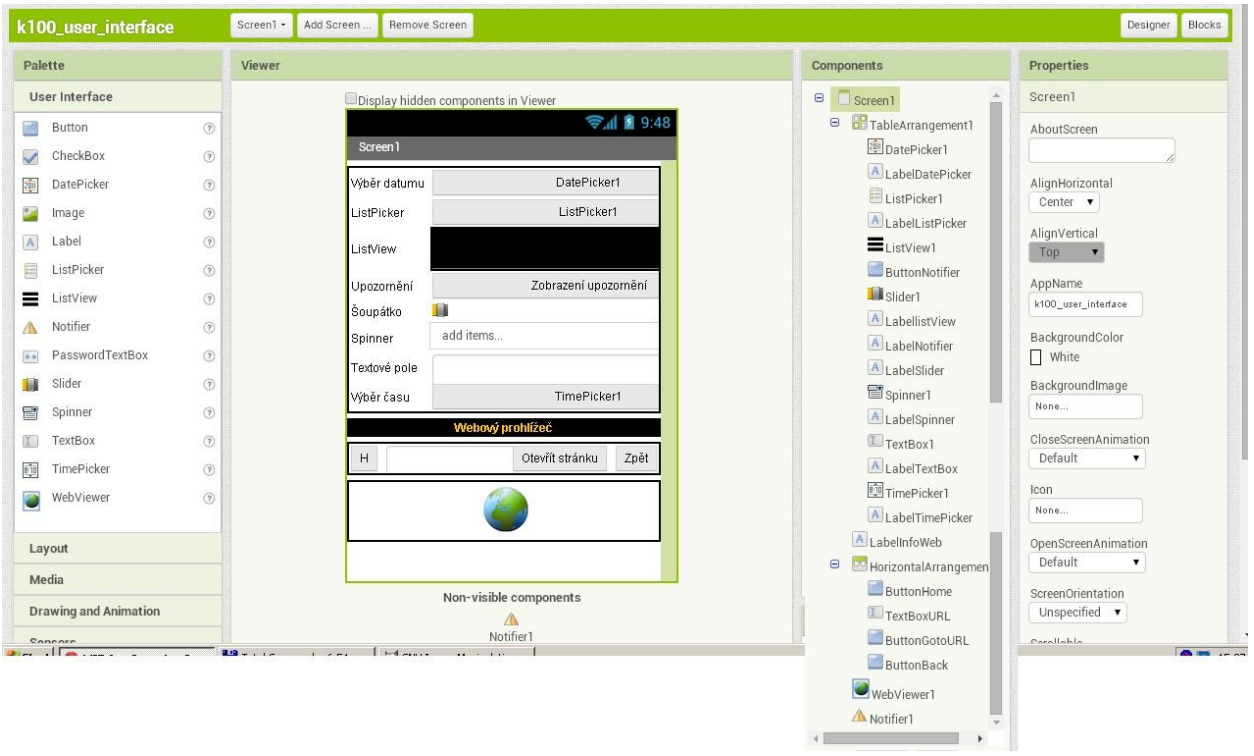
User Interface. (viz. on-line kurz)

Video-návod, jak pracovat s komponentou How to work with listpicker. (viz. on-line kurz)

Video-návod, jak vytvořit vlastní webový prohlížeč. Ai2 app Inventor tutorial WebView. (viz. on-line kurz)

Ukázková aplikace prvků z nabídky User Interface a WebViewer

V této apliakce je ukázka všech prvků z User Interface, které jsme zatím v nepoužívali. Pro každý prvek je doplněn Label, který reaguje na uživatelskou akci. V dolní části je komponenta WebViewer, do které budeme zobrazovat obsah webových stránek. Screen1 má povolenu vlastnost Scrollable.

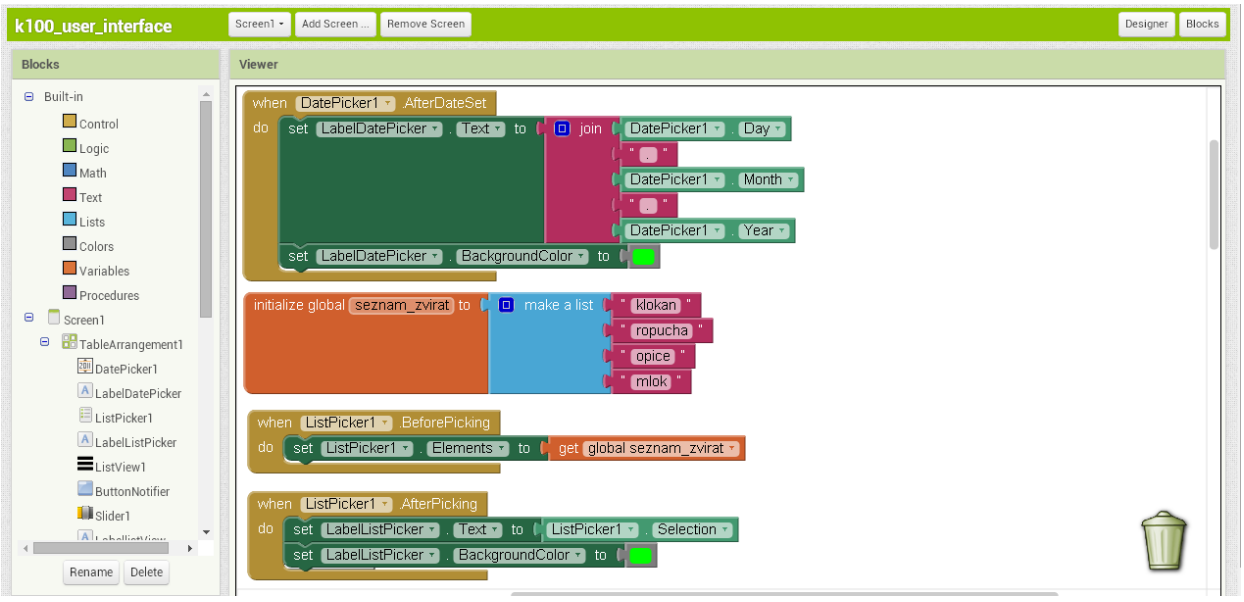


Bloky ukázkové aplikace

Výběr data (den, měsíc, rok) reaguje na událost dokončení výběru. V tu chvíli se složí text, který je vložen do popisku LabeldataPicker. Popiskům, které byly použity pro výstup, je nastaveno zelené pozadí.

Následuje vytvoření testovacího seznamu.

U výběrového seznamu ListPicker1 obsluhujeme dvě události. BeforePicking, ve kterém komponentu propojíme se seznamem. A AfterPicking, ve které předáme vybranou položku do textu popisku.

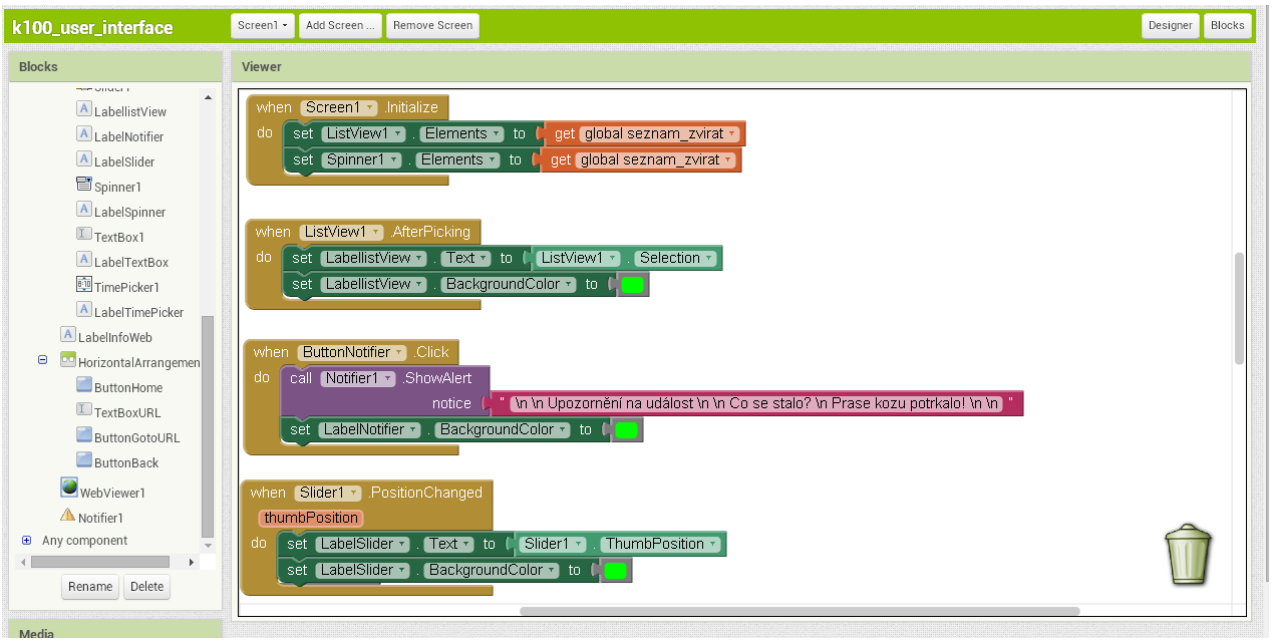


Bloky ukázkové aplikace

Některé komponenty potřebují inicializaci. Tu můžeme provést například v bloku when Screen1.Initialize. V našem případě inicializujeme komponenty ListView1 a Spinner1 naším testovacím seznamem. Z komponenty ListView1 pak při události AfterPicking předáváme hodnotu vybrané položky do popisku. Spinner1 je ukázán později.

S obsluhou upozornění "notifikace" se pracuje jiným způsobem. Po stisknutí tlačítka ButtonNotifier je zavolána procedura upozornění, které je předán řetězec jako obsah zprávy. Zobrazení upozornění je závislé na systému. Většinou jde o vrstvu, která se zobrazí nad plochou aplikace.

Šoupátko "slider" umožňuje výběr z předem definovaného rozsahu hodnot. Reaguje na změnu pozice a aktuální hodnotu předává do popisku.

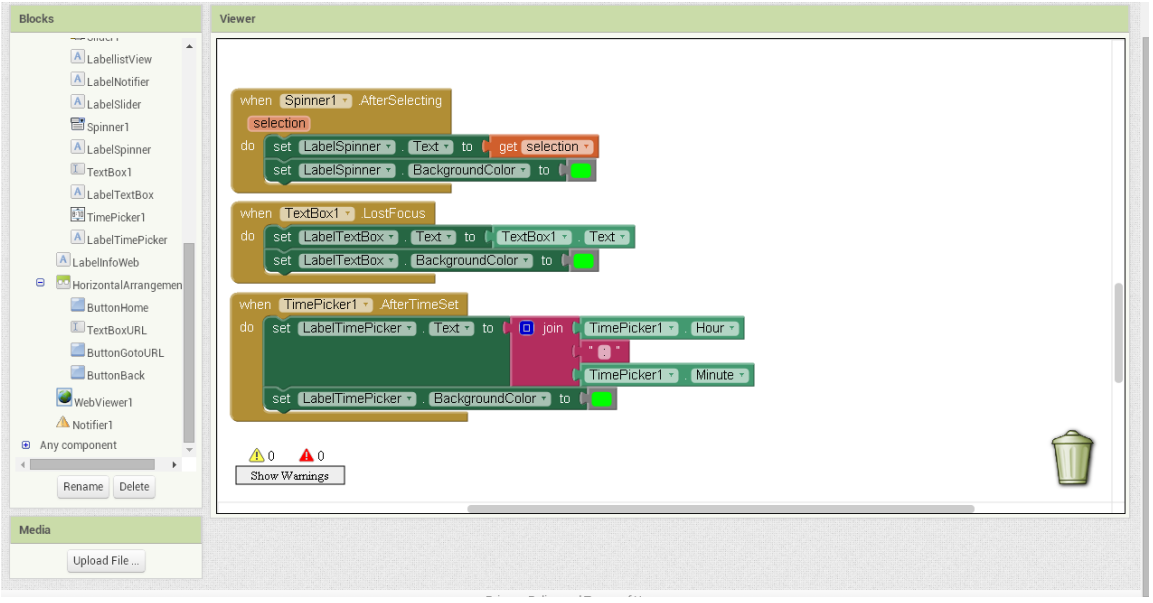


Bloky ukázkové aplikace

Spinner1 je rotující výběrový seznam. Po výběru předá hodnotu vybrané položky popisku.

TextBox1 patří k základním nástrojům komunikace s uživatelem. V našem případě probíhá předání hodnoty při události LostFocus, tedy při ztrátě výběru. K této události dochází, pokud jiná komponenta získá Focus. V naší modelové aplikaci většina komponent Focus nepoužívá, a tak nemusí být snadné tuto akci vyvolat.

Výběr času je komponenta velmi podobná výběru data (den, měsíc, rok), i obsluha je podobná.

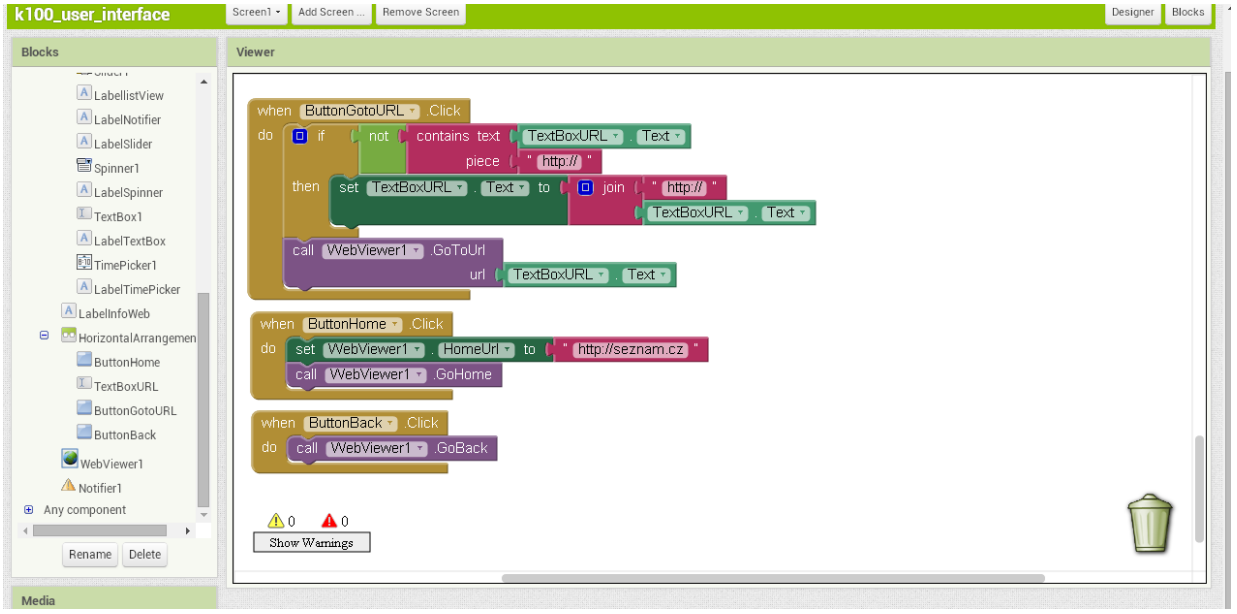


Bloky obsluhy komponenty WebViewer

K webovým stránkám neodmyslitelně patří prohlížeče webových stránek. Jak by bylo možné vyrobit vlastní prohlížeč, když na tak náročném úkolu se podílí tisíce programátorů? Velmi snadno, pokud využijeme připravené komponenty, která většinu funkcí obslouží ve své základní výbavě.

Pro výrobu vlastního prohlížeče potřebujeme pouze zaznamenat a předat URL adresu webové stránky, kterou má prohlížeč načíst a zobrazit. To řešíme pomocí textového pole TextBoxURL. Je zde doplněna kontrola, zda adresa stránky obsahuje řetězec http://, a pokud ne, doplní ho. Pak je adresa stránky předána proceduře komponenty WebViewer1, která se postará o načtení a zobrazení stránky.

Ukázkový prohlížeč je doplněn o tlačítko s přechodem na domovskou stránku. Podobným způsobem by bylo možné vyrobit tlačítka pro přímý přechod na jiné stránky. Protože by v případě použití systémového tlačítka Zpět byla vypnuta naše aplikace, nelze ho využít pro přechod zpět v prohlížeči. Proto je prohlížeč doplněn o tlačítko zpět, které lze použít
k cestě zpět navštívenými stránkami.



Tipy na další práci:

vytvořit si samostatnou aplikaci s prohlížečem stránek a vyřešit snadný přechod na X oblíbených stránek, rozšířit vlastní aplikaci o některé komponenty z User Interface,
vyzkoušet pomocí slideru ovládat parametry některých komponent, např zvětšování / zmenšování obrázku.

14 Závěrem

Co dál?

V čase, který můžete věnovat další tvorbě, doporučuji věnovat čas vašim vlastním programům. Na programech můžete pracovat samostatně, lépe však ve skupinkách. Tvorba programů není jen o programování, je to také o návrhu aplikace, plánování a dalších činnostech, které patří spíše k softwarovému inženýrství.

Doufám, že vám programovací prostředí App Inventor a tento kurz pomohly nahlédnout pod pokličku programování pro prostředí Andoid. Pro některé z Vás to možná bude jediný a dostačující způsob, jak pro Android vytvářet programy, pro některé z vás to bude jen jeden z prvních kroků, díky kterým si ochočíte komplikovanější vývojová prostředí.

Děkuji za společnou procházku tímto kurzem.

Ať už se vaše kroky budou ubírat různými cestami, ať je vaše putování prospěšné a radostné.

Filip Vaculík

Příloha

Odkazy

Odkazy na MIT App Inventor
Hlavní stránka vývojového prostředí MIT App Inventor: http://appinventor.mit.edu/
Odkaz do vlastního vývojového prostředí: http://ai2.appinventor.mit.edu/
Popis charakteristik a použitých konceptů AI2: http://appinventor.mit.edu/explore/ai2/concepts.html
Odkaz na databázi ukázkových aplikací: http://appinventor.mit.edu/explore/ai2/tutorials.html
Sborník velkého množství ukázkových kódů snippets. http://puravidaapps.com/snippets.php
Informace o aktualizacích vývojového prostředí. http://appinventor.mit.edu/ai2/ReleaseNotes.html

Odkazy na informace v angličtině
Android na Anglické Wikipedii: http://en.wikipedia.org/wiki/Android_(operating_system)
Hlavní stránka Androidu: http://www.android.com/
Hlavní databáze aplikací pro Android: https://play.google.com/store

Odkazy na informace v češtině
Android na České Wikipedii: http://cs.wikipedia.org/wiki/Android_(operační_systém)
Hlavní české stránky Androidu: http://www.svetandroida.cz/
Český slovník výrazů a zkratek týkajících se Androidu: http://slovnik.androidforum.cz/
České diskuzní fórum: http://androidforum.cz/
Česká odnož Play Store: http://www.androidmarket.cz/

Odkazy na pomocné nástroje
Emulátor mnoha typů zařízení pro Android: http://www.genymotion.com/
Hlavní vývojové prostředí pro Android: http://en.wikipedia.org/wiki/Android_Studio

Práva k použitým materiálům

Koláž, logo kurzu:
- Logo MIT app inventor (CC)
- Logo Android a nápis Android
- Reklamní fotky mobilů
- Vlastní screenshot kódu

Test, obrázek loga projektu Replicant
Test, výseč grafu zveřejněného pod licencí (CC BY-SA 3.0).

Vlastní tvorba (vydáno s licencí CC Attribution 3.0 Unported):
- Zdrojové kódy ke všem ukázkovým programům.
- Zvuk Hello World v první lekci, záznam strojového čtení.
- Obrázek Hello World v první lekci.
- Screenshoty obrazovek z prostředí App Inventoru.
- Screenshoty z prostředí emulátorů.
- Screenshoty z prostředí OS.
- Kliparty k tématu vesmír a z nich vyexportované obrázky.
- Kliparty skupiny symboly a z nich vyexportované obrázky.

Loga, značky firem, značky produktů jsou užívány pouze v přímé souvislosti s původními vlastníky a užití v tomto kontextu není omezeno zákonem § 8/1 zákona o ochranných známkách.

Cat_illustration.jpg Jan Gillbank, Creative Commons Attribution 3.0 Unported

Odkaz na licenční podmínky:
CC BY-SA 3.0
CC Attribution 3.0 Unported
(odkazy viz. on-line kurz)

ANDROID

Programování v prostředích s podporou jazyka C#, vývoj aplikací založených na animaci a dalších technikách

Projekt se zaměřením na různé, méně tradiční, přesto funkční a uživatelsky přívětivé techniky vývoje Android aplikací.

Využité přístroje a programy:

- osobní počítač (notebook)
- mobilní dotykové zařízení (tablet, dotykový telefon)
- emulátor BlueStacks
- SharpDevelop
- dot42
- Xamarin
- Adobe Flash Professional
- Stencyl

Cílová skupina/náročnost: 1. až 4. ročník SŠ a odpovídající ročníky gymnázií

Všechny uvedené texty, obrázky a videa jsou vlastní, není-li uvedeno jinak. Autory Youtube embed videí lze nalézt při kliknutí na znak Youtube ve videu během přehrávání.

Autoři:

Mgr. Denis Mainz, Mgr. Jan Hodinář

K plnohodnotnému využití této studijní opory je nutný přístup k on-line zdrojům a materiálům.

Tento materiál vznikl z finanční podpory Evropského sociálního fondu a státního rozpočtu České republiky v rámci projektu „Popularizace vědy a badatelsky orientované výuky“, reg .č. CZ.1.07/2.3.00/45.0007.

TVORBA MOBILNÍ APLIKACE V JAZYCE C#

1 Základní informace o projektu

Název:

Tvorba mobilní Android aplikace v jazyce C#

Anotace programu/zaměření/hlavní cíl:

Cílem projektu je zvládnout postup při tvorbě mobilní Android aplikace využívající programovací jazyk C#.

Cílová skupina:

žáci 2. až 4. ročníků středních škol a odpovídajících ročníků gymnázií

Organizační podmínky

Samostatná práce studentů. (1 student na 1 počítač/tablet)

Pomůcky

Počítač, sada vývojářských nástrojů pro C# a Android, tablet (případně chytrý telefon nebo emulátor Androidu)

Časová náročnost

(max. 2×45 minut)

Vazba na RVP

RVP pro Gymnázia: Vzdělávací oblast Informatika a informační a komunikační technologie

RVP pro SŠ obory L0 a M(obor 18-20-M/01 Informační technologie): Vzdělávací oblast Vzdělávání v informačních a komunikačních technologiích

Mezipředmětové vazby

V závislosti na jednotlivých aktivitách

Fáze projektu

- Seznámení se s technologiemi .NET a Android
- Volba mezi možnostmi integrace C# aplikací v operačním systému Android Instalace vývojářských nástrojů pro integraci jazyka C#
- Tvorba C# aplikací, jejich následný export do zařízení s OS Android Prezentace vlastních aplikací
- Hodnocení

2 Motivační rámec projektu

Proč C# na Androidu?

Umíte programovat na platformě .NET a zoufáte si, že si na svůj nový tablet nebo chytrý telefon nemůžete naprogramovat žádnou aplikaci, protože běží na Androidu? Chcete se jako programátor prosadit na trhu s mobilními aplikacemi, ale nechcete se učit Javu nebo jiný nový jazyk, abyste toho docílili? Nechce se vám moc platit za drahé mobilní aplikace, které byste si pro vlastní potřebu udělali sami a lépe?
Myslíte, že je to nemožné? Nezoufejte. Jde to! Stačí chvilku studovat a za pár hodin budete schopni tvořit svoje vlastní aplikace pro Android psané v C#.

Naučná videa s tematikou C# na Androidu (viz. on-line kurz):

How To Develop Android Apps with C#

C# Android Development | GUI & First App | Part 1

C# Android Development | GUI & First App | Part 2

Zajímavé stránky a jiné studijní materiály (viz. on-line kurz):

Mono for Android - Create amazing Android apps with C# and .NET (e-kniha)

Stránka vývojářského nástroje dot42

Stránky platformy Xamarin

3 Poznámky k využití přístrojů

Co budeme potřebovat?

K tomu, abychom mohli úspěšně vyvíjet aplikace pro Android psané v C#, budeme potřebovat několik věcí: Osobní počítač nebo notebook, tablet nebo chytrý telefon, vývojářské nástroje a případně také emulátor operačního systému Android.

PC

Jakýkoliv počítač nebo notebook současné generace, na který se nainstalují vývojářské nástroje (a případně emulátor) a na který se případně připojí tablet nebo chytrý telefon. Nutný je operační systém Windows a nainstalovaná nejnovější platforma .NET

Tablet

Tablet nebo chytrý telefon s operačním systémem Android (je jedno, jakou má verzi jádra OS). Tento tablet může být připojen k počítači během vývoje a programátor si na něm může rovnou zkoušet ladit dotyčnou aplikaci. V případě, že tablet nemáme k dispozici, je třeba na PC nainstalovat emulátor operačního systému Android a testovat to na něm.

Vývojářské nástroje

dot42 C# for Android Xamarin

V aktivitě Jak dostat C# na Android se dozvíme všechny možné způsoby, jak implementovat jazyk C# na operační systém Android. Podle toho, jakou možnost zvolíme, je třeba stáhnout a nainstalovat vývojářské nástroje nebo aplikace, které nám to umožní. Zde jsou odkazy na dvě nejpravděpodobnější volby, a to jsou dot42 a Xamarin.

Stránky projektu dot42

Stránky projektu Xamarin
(odkazy viz. on-line kurz)

Emulátor

BlueStacks

V případě, že nevlastníme nebo po dobu vývoje nemáme k dispozici tablet nebo chytrý telefon s operačním systémem Android, je třeba si na PC nainstalovat jeho emulátor. Těch existuje celá řada a dokáží zastoupit fyzické zařízení ve všech směrech. Dokonce mají tu výhodu že většinou podporují speciální funkce pro vývojáře, které "ostrá" verze nemá. Zde je odkaz k jednomu z nejznámějších, který se dá pořídit zadarmo, a to je BlueStacks.

Stránky emulátoru BlueStacks (odkaz viz. on-line kurz)

4 Použité technologie

Téma	Použité technologie	
Tematický celek	Tvorba mobilní Android aplikace v jazyce C#	
Motivační rámec	Abychom se mohli pustit do praktické části vývoje aplikací pro Android v C#, musíme nejprve získat základní fakta o těchto technologiích a pochopit, proč původně nejsou kompatibilní.	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	PC a internet	
Stručný popis aktivity s využitím	Studenti s využitím PC a internetu nastudují fundamentální základy technologií Android a .NET.	
Vhodné místo	Běžná počítačová učebna s přístupem k internetu.	
Cíle aktivity	Žáci budou schopni pochopit a vysvětlit základní vlastnosti technologií Android a .NET.	
Rozvíjené	Kompetence k učení	
Předchozí znalosti	Práce s počítačem, vyhledávání na internetu.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Studium problematiky Android a .NET	Samostatná práce, samostudium a následná diskuse vázaná na studovanou oblast
Hodnocení	Hodnocení proběhne až po praktické části.	
Návaznosti	Aktivita Jak dostat C# na Android	

Zadání:

1. Nastudujte základní informace o technologiích Android a .NET.

2. Pokuste se zdůvodnit, proč aplikace na platformě .NET nejsou pro Android v základu vhodné.

Teorie:

V této kapitole jsou klíčové dva základní kameny potřebné pro implementaci aplikace v jazyce C# na OS Android. Prvním je samotný OS Android a druhým je platforma .NET.

.NET

.NET je skupina technologií vyvinutých společností Microsoft, které dohromady tvoří celou platformu. Tato skupina technologií je open source (open source – otevřený zdrojový kód, který lze prohlížet či upravovat) a je určena pro výrobu nových aplikací pro operační systém Windows. Skládá se ze dvou hlavní částí, a to FCL (Framework Class Library) a CLR (Common Language Library), které dohromady tvoří rozhraní .NET Framework. FCL je knihovna tříd, která poskytuje jazykovou interoperabilitu, jinak řečeno tento kód lze použít v jiných jazycích. CLR je virtuální stroj, který poskytuje služby jako např. bezpečnost, správu paměti či zpracování výjimek. Platforma je dále poskytována v dalších dvou verzích, a to pro mobilní telefony a zařízení s omezenými zdroji. Mobilní verze (.NET Compact Framework) je k dispozici na chytrých telefonech nebo mobilních telefonech se systémem Windows Mobile. Pro platformu jsou využívána hlavně vývojová prostředí Visual Studio(placené) či SharpDevelop(freeware licence).

Historie

Vývoj platformy .NET započal rokem 1990 pod názvem Windows Services, do roku 2000 byla vyvinuta první beta verze .NET verze 1.0. První plná verze byla vydána začátkem roku 2002 s vývojovým nástrojem Visual Studio. Od verze 3.0 (od roku 2006) je .NET Framework součástí systémů Windows, avšak do verze s Microsoft Windows XP nebyl zahrnut.

- .NET Framework 1.0 – vydána 13. února 2002, podpora skončila 14. července 2009, uvedena v systémech Windows 98, ME, NT 4.0,
  2000 a XP
- .NET Framework 1.1 – vydána 3. dubna 2003, součástí vydání Visual Studio, první verze, která byla zahrnuta do systému (Windows Server 2003), podpora skončila 8.října 2013, poslední podpora Windows NT 4.0
- .NET Framework 2.0 – vydána 22. ledna 2006, vydáno spolu s Visual Studio 2005 a Microsoft SQL Server 2005, podpora 64-bitových operačních systémů, poslední podpora Windows 98 a Windows ME
- .NET Framework 3.0 – vydána 21. listopadu 2006, dodávána se systémem Windows Vista a Windows Server 2008 jako volitelná součást, podpora 3D grafiky a Direct3D technologií
- .NET Framework 3.5 – vydána stejně jako .NET Framework 3.0, podpora Windows Mobile a Windows CE
- .NET Framework Service Pack 1 – lepší výkon za určitých podmínek, vydána 11. srpna 2008, dodávána s Windows 7 a Windows Server 2008 jako volitelná část

- .NET Framework 3.5 SP1 Client Profile – instalace pouze komponent, které jsou nejdůležitější pro desktopové aplikace
- .NET Framework 4 – vydána 12. dubna 2010 ve verzi s Visual Studiem 2010, cílem byla podpora vícejádrových procesorů, finální verze 4.0.3 vydána 4. března 2012
- .NET Framework 4.5 – vydána 15. srpna 2012, vylepšené funkce, podpora pouze pro Windows Vista
- .NET API – podpora pro aplikace METRO (grafické rozhraní Windows 8), podpora rozhraní .NET Framework, C++, HTML 5, JavaScript, finální verze 4.5.2 podpora Windows 8.1

Architektura

1. **CLI (Common Language Infrastructure)** - účelem je poskytnout neutrální jazykovou platformu pro vývoj a spuštění aplikací, jinak řečeno nebude vázána na jeden jazyk, ale bude k dispozici v rámci několika jazyků podporovaného rámce
2. **CIL (Common Intermediate Language)** - kód, jenž je umístěn v CLI sestavách, skládá se z jednoho nebo více souborů, obsahuje manifest, který má data pro výrobu, obsahuje název sestavy, textové jméno, číslo verze a veřejný klíč
3. **Bezpečnost** - .NET má svůj vlastní bezpečností mechanismus, který má dvě části: zabezpečení přístupu ke kódu(CAS), validaci a verifikaci
4. **Knihovna tříd** - .NET obsahuje standardní sadu knihoven: knihovny FCL a základní sadu knihoven (BCL), FCL zahrnuje podmnožinu celé knihovny tříd, které slouží jako základní API z CLR, např. třídy system.dll a system.core.dll jsou součástí FCL, BCL je podmnožinou FCL a vztahuje se na celou třídu knihoven dodávaných s .NET Framework, obsahuje sadu včetně Windows Forms, ADO.NET, ASP.NET, LINQ, Windows Presentation Foundation a Windows Communication Foundation
5. **Správa paměti** - osvobozuje vývojáře od řízení paměti (přidělování a uvolňování), .NET obsahuje tzv. Garbage collector, který běží na samostatném vlákně, který vyčte všechny nepoužitelné objekty a kultivuje k nim přidělené.

Jazyk C#

Jazyk C Sharp (vyslovuje se "sí šarp") je objektově orientovaný programovací jazyk vyvinutý společností Microsoft jako součást platformy
.NET. Díky tomu je jedním z programovacích jazyků navržených přímo pro CLI (Common Language Infrastructure). Nedlouho po jeho vzniku byl standardizován organizacemi Ecma (ECMA-334) a ISO (ISO/IEC 23270:2006). Při jeho tvorbě bylo dodržováno několik specifických pravidel, která ho nadále charakterizují:

- Jazyk C# je zamýšlen jako jednoduchý, moderní, obecně zaměřený, objektově orientovaný programovací jazyk.
- Jazyk a jeho implementace by měly zajistit podporu pro moderní principy softwarového inženýrství, jako jsou silné kontroly datových typů, kontrola hranic polí, detekce použití neinicializovaných proměnných a samočinné uvolňování paměti (Garbage collection).
  Softwarová robustnost, odolnost a programátorská produktivita jsou také důležité.
- Jazyk je zamýšlen pro vývoj softwarových komponent vhodných pro nasazení v distribuovaných systémech.
- Portabilita zdrojového kódu je velmi důležitá, stejně jako portabilita (přechod z jazyka na jazyk) programátorů, zvláště těch, kteří již znají jazyky C a C++.
- Podpora jazykových mutací softwaru je velmi důležitá.
- C# je zamýšlen jako vhodný jak pro hlavní i vnořené systémy, které jdou od velmi obsáhlých aplikací využívajících sofistikovaných operačních systémů až po malé jednoúčelové aplikace se specifickými funkcemi.
- Ačkoliv je jazyk C# zamýšlen jako velmi ekonomický s ohledem na využívání paměti a procesorového času, není zamýšlen jako přímá konkurence vzhledem k výkonu aplikací psaných v jazyce C a C++.

Operační systém Android

Android je mobilní operační systém vyvinutý společností Google, který je založen na linuxovém jádře. Primárně je určen pro zařízení s dotykovou obrazovkou (smartphony, tablety, hodinky, chytré televizory atd.). Android je nepoužívanější mobilní OS od roku 2013. Aplikace pro tento operační systém lze stahovat pomocí obchodu/úložiště Google Play. Zdrojový kód Androidu je volně dostupný pod open source licencí.

Historie

Android byl založen v říjnu roku 2003 v Kalifornii, zakladateli jsou Andy Rubin, Rich Miner, Nick Sears a Chris White. Prvotním záměrem bylo rozšířit konkurenci pro operační systémy Symbian (Nokia) a Windows Mobile (Microsoft). Google získal Android roku 2005 a následně vytvořil tým okolo Rubina za účelem výroby mobilní platformy vytvořené na linuxovém jádře. Spekulace o tom, kdy má Google vstoupit na trh, se objevily roku 2006. Prototyp pod krycím názvem „Sooner“ měl podobnost se zařízeními Blackberry, bez dotykového displeje. Později byl inovován, aby konkuroval firmám LG a Apple. Koncem roku 2007 bylo vytvořeno sdružení několika firem (HTC, Sony, Samsung, Google, T-Mobile atd.), které představilo cíl o otevřených standardech na

mobilních zařízeních. Tentýž den byl představen první smartphone postavený na jádře Linuxu ve verzi 2.6.25. První komerčně dostupný telefon byl HTC Dream roku 2008. Od roku 2008 Android vydal řadu aktualizací, které postupně zlepšovaly OS, nejnovější verze Android v současné době (konec roku 2014) je verze Android 5.0, též nazýván
„Lollipop“. V roce 2010 Google spustil výrobu přístrojů Nexus, což je řada smartphonů a tabletů se

systémem Android.

Vlastnosti

Hlavní vlastností Androidu je to, že je založen na přímé manipulaci pomocí dotykových vstupů, které následně odpovídají reálným akcím. Reakce na vstup je navržena tak, aby pomocí dotykového rozhraní a vibračního rozhraní poskytla hmatovou zpětnou vazbu uživateli. Zařízení jsou vybavena několika senzory, akcelerometry, gyroskopy a jinými čidly. Hlavní obrazovka, též nazývána jako domovská, se vyznačuje ikonami aplikací a widgety (aplikace spuštěná na hlavní obrazovce), můžeme rolovat mezi několika dalšími obrazovkami, které si uživatel může nastavit. Většina zařízení běžících na OS Android se vyznačuje hlavními rysy, kterými jsou spodní lišta se 3 tlačítky pro návrat na hlavní obrazovku, zpětné tlačítko a tlačítko se spuštěnými aplikacemi, které jsou na pozadí. V zařízení jsou umístěny diody (podle typu výrobce), které uživatele upozorňují na zmeškané události jako např. příchozí SMS, volání, vybití nebo nabíjení baterie.

Aplikace a jejich vývoj

Aplikace, které rozšiřují funkčnost zařízení na OS Android, jsou psány především v programovacím jazyce Java pomocí nástroje pro vývoj softwaru pro Android (SDK). SDK obsahuje několik vývojových nástrojů včetně debuggeru, softwarové knihovny, emulátoru založeného na QEMU, dokumentace, ukázkových kódů a návodů. Oficiálně podporované vývojové prostředí je Eclipse pomocí ADT pluginů (Android Development Tools). Další vývojové nástroje jsou např. Google App Inventor či různé multiplatformní mobilní webové aplikace. Primárním úložištěm aplikací je již zmíněný Google Play.

Bezpečnost

Tento OS má několik druhů zabezpečení, ať už se jedná o uzamykaní obrazovky či SIM karty, nebo samotného telefonu pomocí softwaru. V nynější době jsou vyvíjeny aplikace zaměřené na vyhledání ztraceného telefonu pomocí lokace GPS. Podle kritiků chybí v Androidu podpora šifrování či podpora ochrany proti malwaru. Android běží v tzv. izolovaném prostředí, kde uživatel nemá přístup k souborovým systémům. Před instalací samotného softwaru (appky) přes Google Play zobrazí všechna potřebná oprávnění k instalaci. Nejčastější formou zneužívání OS jsou pak textové zprávy, kde jsou SMS odesílány na tzv. prémiová čísla bez souhlasu či oprávnění uživatele.

5 Jak dostat C# na Android

Téma	Jak dostat C# na Android	
Tematický celek	Tvorba mobilní Android aplikace v jayzce C#	
Motivační rámec	Když už víme, že technologie .NET a OS Android pro sebe nikdy nebyly určeny, chápeme, že implementace jazyka C# v OS Android může být problém, je tedy třeba nastudovat způsoby,	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	PC a internet	
Stručný popis aktivity s využitím přístroje	Studenti s využitím PC a internetu nastudují možnosti implementace C# aplikace na OS Android.	
Vhodné místo	Běžná počítačová učebna s přístupem k internetu.	
Cíle aktivity	Studenti budou schopni vybrat způsob implementace C# aplikace na OS Android.	
Rozvíjené kompetence	Kompetence k učení	
Předchozí znalosti	Aktivita navazuje na aktivitu Použité technologie	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Studium technologií umožňujících implementaci aplikace psané v jazyce C# na OS Android.	Samostatná práce nebo práce ve skupině s diskusí
Hodnocení	Hodnocení proběhne až po praktické části.	
Návaznosti	Aktivita Instalace a nastavení vývojářských nástrojů dot42	

Zadání:

1. Nastudujte možnosti implementace C# aplikace na OS Android.

2. Proberte klady a zápory jednotlivých možností a jednu zvolte.

Teorie:

dot42 C# for Android Vs Xamarin

Jak už předchozí kapitola napovídá, drtivá většina vývojářů pro OS Android používá pro své aplikace programovací jazyk Java. Nicméně i vývojáři znalí pouze platformy .NET mají šanci vyvíjet plnohodnotné aplikace na Android. Než však učiní první krok, je třeba zvolit metodu, pomocí níž k tomuto problému budou přistupovat. Volby jsou totiž víceméně dvě a každá z nich, jak to bývá, skýtá některá svoje úskalí. V této části nebudeme technologie představovat zvlášť, ale aby si začínající programátor mohl lépe srovnat sám, co si od své budoucí kariéry v této oblasti představuje, postavíme tyto technologie proti sobě v několika klíčových oblastech. Jelikož programátoři se někdy od sebe značně liší a každý má jiné zkušenosti a priority, je docela možné, že volba bude pro každého jiná.

Nástroje Vs Platforma

Jak vidíme na následujících ukázkách kódu, obě dvě "technologie" užívají jazyk C# přímo v

aplikacích pro OS Android: **dot42:**

```
[Activity(Icon = "Icon", Label = "dot42 Using Button!")]
public class MainActivity : Activity
{
  protected override void OnCreate(Bundle savedInstance)
  {
    base.OnCreate(savedInstance);
    SetContentView(R.Layouts.MainLayout);
    var myButton = FindViewById<Button>(R.Ids.myButton);
    myButton.Click += (s, x) => myButton.Text = "Hi!";
  }
}
```

Xamarin:

```
public async Task<List<FeedItem>> GetFeedItems(DateTime date) {
  var feed = "http://planet.xamarin.com/feed/";
  var response = await httpClient.GetStringAsync(feed);
  var items = await ParseFeedAsync(response);
  return items.Where(item => item.Published.Date == date).ToList();
}
```

Každá z těchto technologií však přistupuje k řešení problému jinak a díky tomu vznikají někdy až diametrální odlišnosti, které mají vliv nejen na programátora a proces programování, tak i na samotnou výslednou aplikaci. Zatímco dot42 je spíše sada vývojářských nástrojů umožňující kompilaci C# kódu do zdrojového kódu OS Andorid (DEX), firma Xamarin dodává celou masivní platformu pro vývoj mobilních aplikací, a to nejen na OS Android, ale společně také na iOS. To má některé dalekosáhlé dopady, které rozebereme dále.

Velikost výsledné aplikace
Díky tomu, co bylo naznačeno výše, se výsledné aplikace liší hlavně z hlediska velikosti. V tomto ohledu má nevýhodu platforma Xamarinu. Aplikace Xamarinu mají v sobě zabudovaný takzvaný "overhead", což je část kódu, která zajišťuje, aby samotná aplikace na zařízení mohla vůbec běžet (aby jí zařízení rozumělo a bylo schopné ji spustit tak, jak je napsaná a zkompilovaná). Podobný overhead mají i flashové aplikace napsané pomocí Adobe AIR, který se tomu však může vyhnout v případě, že si adobe AIR uživatel na zařízení nainstaluje samostatně. U Xamarinu z toho pak vyplývá to, že jeho aplikace jsou v průměru o několik megabajtů větší než aplikace nativní nebo psané v dot42. To se zdá jako vcelku málo, ale tento objem dat roste, čím víc aplikace používá API funkcí. To ve výsledku způsobuje delší stahování aplikací a vetší místo nutné k jejich skladování na zařízení.

Vývojové prostředí

Aplikace pro vývoj samotných mobilních aplikací nebo i obecně aplikací nemobilních se nazývá vývojové prostředí. Ve většině případů je zaměřeno na jeden programovací jazyk, ovšem existují i taková prostředí, které lze pomocí pluginů "zaměřit"na více jazyků. Když prozkoumáme programátorskou komunitu, věnující se jazyku C#, zjistíme, že v tomto ohledu převládají dvě aplikace. Volně šiřitelné prostředí SharpDevelop a komerční Visual Studio přímo od Microsoftu. Prostředí SharpDevelop je ve srovnání s Visual Studiem o dost jednodušší a má mnohem méně funkcí, proto ho často používají školy a programátoři začátečníci, aby se dostali do problematiky programování v .NET. Když však programátor chce pracovat komerčně nebo je součástí nějaké firmy, v drtivé většině případů sáhne sice na relativně drahé, ale velmi kvalitně zpracované Visual Studio. Většina programátorské komunity se pak obecně shodne na tom, že Visual Studio je jedním z nejlepších programátorských nástrojů vůbec.

Pokud se v tomto ohledu podíváme na naše dvě volby, vznikne nám jakási remíza. Obě dvě řešení podporují Visual Studio, i když Xamarin ho podporuje pouze v jedné ze svých cenově dražších variant. Dot42 navíc podporuje i SharpDevelop, což je u něj velké plus z pohledu programátorů, kteří s dot42 chtějí pracovat nezávisle a zadarmo. Xamarin však tuto nevýhodu kontruje dostupností svého vlastního vývojového prostředí Xamarin Studio, které je poskytováno ve všech jeho verzích. Toto prostředí je nejen kvalitní, ale je navíc uzpůsobeno přímo vývoji mobilních aplikací v Xamarinu, a proto poskytuje velmi dobré funkce a ergonomii. Nevýhodou je však fakt, že se programátor musí učit pracovat s vývojovým prostředím, se kterým předtím nebyl seznámen.

Grafické rozhraní aplikací

Uživatelský interface (UI) pro Android je u obou řešení prakticky identický. Porovnávat tedy grafiku u obou řešení na Androidu nemá moc smysl. Je však nutno podotknout, že Xamarin jako platforma je univerzálně použitelný nejen pro Android. Tím, že Xamarin nabízí celou platformu, umožňuje implementaci na většinu mobilních operačních systému. Xamarin tedy mimo Android podporuje také iOS a Windows Mobile. To mu v tomto ohledu dává náskok před dot42, protože je pak možné výslednou aplikaci šířit na více druhů zařízení. Co je však v tomto směru nevýhodou, je fakt, že UI na rozdílných zařízeních využívají jiné funkce a API, a proto je nutné aplikaci "přeprogramovat" z hlediska vzhledu pro každý druh zařízení, i když jádro zůstává stejné.

Cena

Důležitým faktorem při výběru našeho řešení pro implementaci C# aplikací na Android je zcela nepochybně také cena. Každá firma přistupuje k ceně poněkud jinak. Nástroje dot42 mají dvě možnosti platby za jejich používání. První z nich je zdarma a je to takzvaná veřejná (community) licence. To znamená, že je možné používat nástroje zadarmo v plné jejich podobě. Je možné aplikace nahrávat do obchodů (Google Play), aby si je mohli uživatelé stahovat, avšak takto vystavené aplikace musí být dostupné zdarma. Pokud vývojáři chtějí na aplikacích začít vydělávat, musí si koupit jednorázovou profesionální licenci, která vyjde na 300 amerických dolarů.

V případě Xamarinu je také dostupná zdarma takzvaná zkušební verze. Tato verze je však značně osekaná. Dovoluje sice nahrávat aplikace do obchodů, ale postrádá značné množství podpory a funkcionality a navíc je velikost výsledné aplikace značně omezena a nutí tak vývojáře, kteří to v tomto směru myslí vážně, koupit si jednu z placených licencí. Placené verze jsou momentálně tři a každá obsahuje o něco více funkcionality a podpory ze strany Xamarinu. Ceny jsou účtovány měsíčně nebo ročně a pohybují se od 25 dolarů za měsíc až po 158 dolarů za měsíc. Aktuální ceník spolu s dostupnou funkcionalitou je možné prohlédnout zde: https://store.xamarin.com/

Závěrečné srovnání

Pokud tedy chceme tyto naše dvě možnosti srovnávat, musíme si nejprve uvědomit, čeho chceme naším vstupem do oblasti mobilních aplikací dosáhnout. Pokud chceme vyrábět menší, levnější aplikace pouze pro OS Android, pak se zdají jasnou volbou nástroje dot42. Pokud však chceme masivní multifunkční aplikace, které budeme chtít šířit na všechny mobilní platformy a zaměříme na to celý náš podnikatelský záměr a celou naši firmu, pak se zdá platforma Xamarin jako lepší odpověď.

Dále se v tomto kurzu budeme zaměřovat již pouze na nástroje dot42, a to z toho prostého důvodu, že je kurz určen převážně pro školy, které většinou nemají peníze na to, aby platily každý měsíc drahé licence jen proto, aby seznámily svoje žáky s možnostmi programování Android aplikací v C#.

6 Instalace a nastavení vývojářských nástrojů dot42

Téma	Instalace a nastavení vývojářských nástrojů dot42	
Tematický celek	Tvorba mobilní Android aplikace v jayzce C#	
Motivační rámec	Dříve než začneme vyvíjet samotné aplikace, je nutné nainstalovat a nastavit vývojářské nástroje, které k tomu účelu budeme používat.	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	PC a vývojářský balíček dot42	
Stručný popis aktivity s využitím přístroje	Studenti na dostupné PC s OS Windows nainstalují a nastaví vývojářské nástroje v balíčku dot42.	
Vhodné místo	Běžná počítačová učebna vybavená operačním systémem Windows.	
Cíle aktivity	Studenti budou schopni nainstalovat a nastavit vývojářský balíček dot42.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů	
Předchozí znalosti	Práce s PC, instalace software	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Zdůvodnění volby vývojářského nástroje dot42	Skupinová diskuse a dialog
30 minut	Instalace vývojářského nástroje dot42	Samostatná práce nebo skupinová práce; skupinová diskuse a dialog
Hodnocení	Bez hodnocení. Studenti budou hodnoceni až po tvorbě vlastních aplikací.	
Návaznosti	Na tuto aktivitu navazuje tvorba první aplikace v C# pro Android "Ahoj světe"	

Zadání:

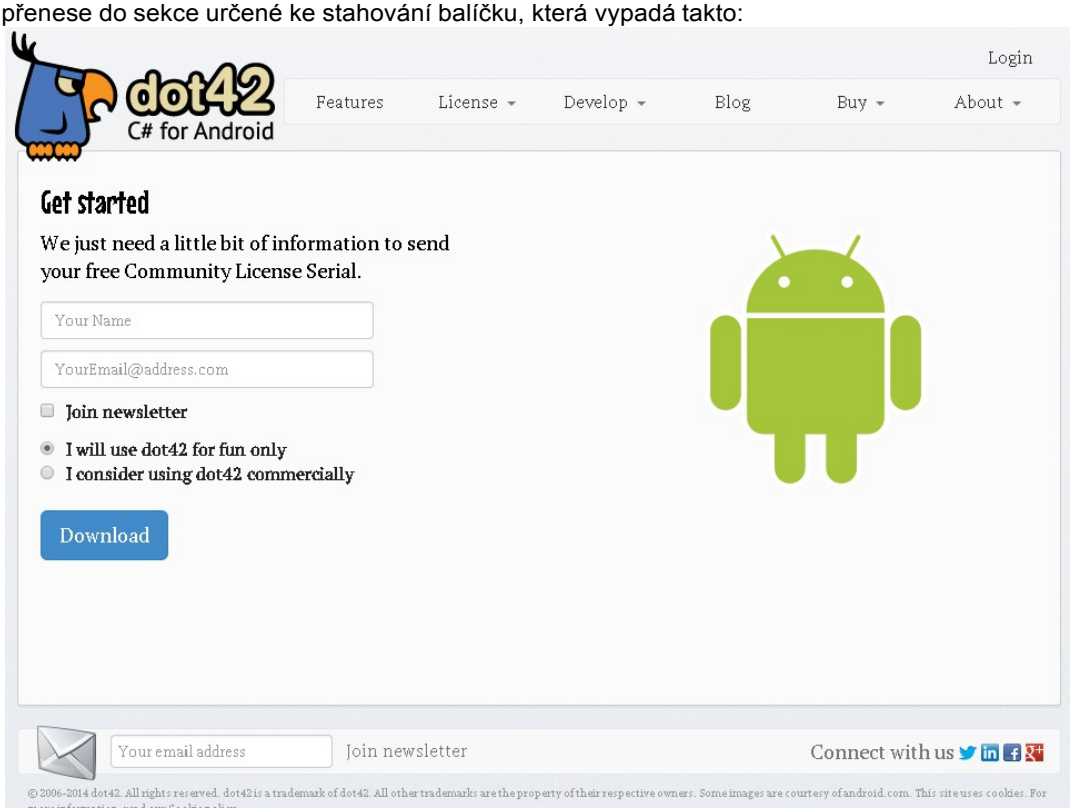
1. Zaregistrujte se a stáhněte si instalační balíček nástrojů dot48.

2. Nainstalujte si a aktivujte nástroje dot48 pro soukromou potřebu.

3. Vyzkoušejte spustit vývojové prostředí SharpDevelop 4.5 nainstalované s balíčkem dot48.

Postup:

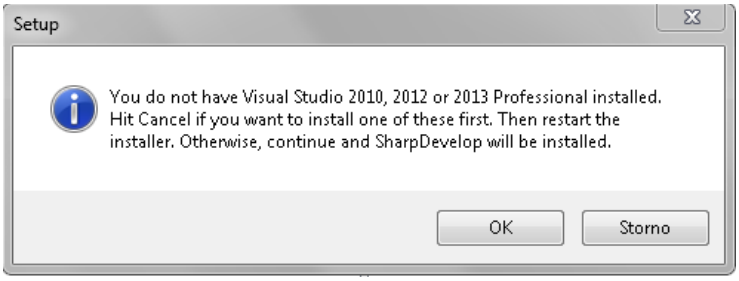
V první fázi je třeba jít na stránku projektu dot48 a zaregistrovat se proto, abychom byly schopni stáhnout jejich instalační balíček. Na adrese https://www.dot42.com/ zvolíme snadno viditelné tlačítko DOWNLOAD, které nás přenese do sekce určené ke stahování balíčku, která vypadá takto:



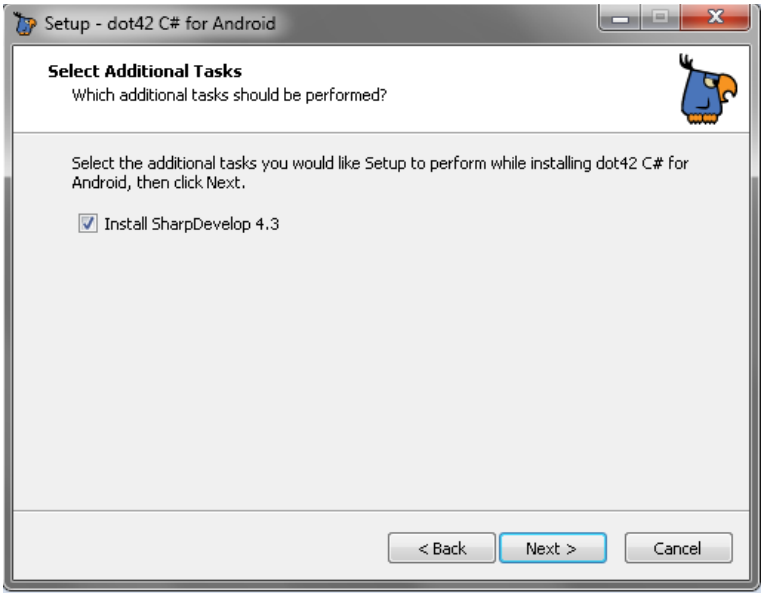
Zde je potřeba vyplnit vaše jméno a e-mailovou adresu. Dejte si pozor, abyste vyplnili adresu správně a k e-mailovému účtu měli přístup. Na tuto adresu vám totiž bude doručen aktivační klíč celého produktu. Dále je třeba zaškrnout, že nebudete používat dot42 pro komerční účely, ale pouze ke své vlastní potřebě (I will use dot42 for fun only). V případě, že byste zvažovali pomocí tohoto produktu vydělávat, je třeba zaplatit značný finanční poplatek ve výši 299$. Pokud nechceme dostávat informace o nových produktech a aktualizacích dot48,

odškrtneme políčko Join Newsletter. Jakmile jsme s vyplňováním hotovi, zmáčkneme Download a uložíme si instalační balíček k sobě do počítače.

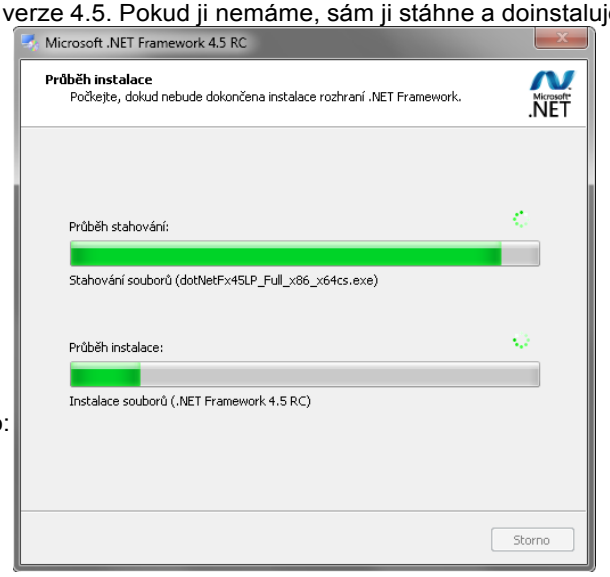
Jakmile máme balíček stažen, spustíme ho pro instalaci. Jako první vám pravděpodobně vyskočí tato hláška:



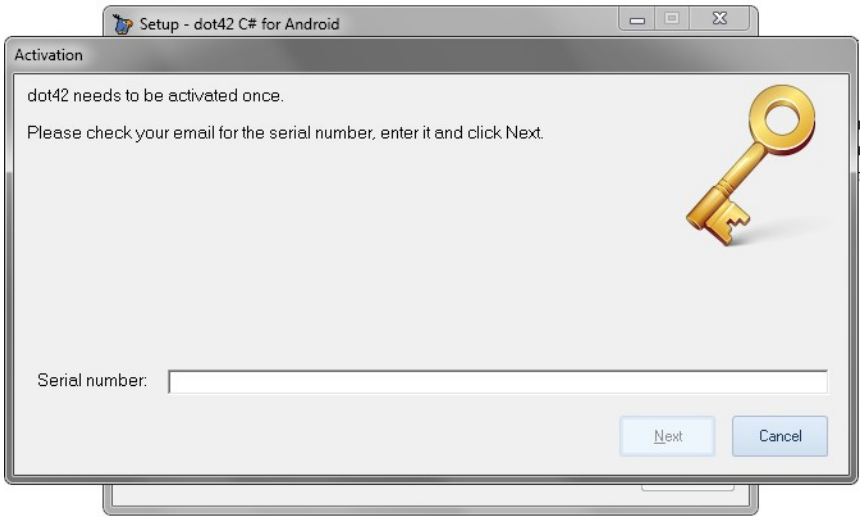
Je to způsobeno tím, že nástroje dot48 pracují s dvěma vývojovými nástroji pro jazyky platformy .NET. První a základní je Microsoft Visual Studio, který je oficální a hlavní, podporovaný společností Microsoft, bohužel je však velice finančně nákladný, i když pro studenty se dá sehnat zdarma. Druhým je volně šiřitelné prostředí SharpDevelop, které je obecně považováno za velmi dobrou alternativu, která je zcela zdarma. Pokud tedy nemáte na počítači nainstalovanou nějakou verzi Visual Studia, objeví se tato hláška a instalace bude pokračovat dál s tím, že vám při instalaci nabídne možnost doinstalovat SharpDevelop, kterou v našem případě využijeme. Jakmile tedy budeme pokračovat v instalaci, objeví se potvrzovací okno, ve kterém necháme zaškrtnutou možnost "instal SharpDevelop 4.3":



Jakmile dáme pokračovat, instalátor zjistí, zda máme potřebnou verzi platformy .NET Framework. V našem případě je to verze 4.5. Pokud ji nemáme, sám ji stáhne a doinstaluje za nás. Průběh takové instalace pak vypadá nějak takto:



Jakmile instalace doběhne, budeme následujícím oknem vyzváni, abychom zadali aktivační klíč:



Tento klíč nám byl zaslán na e-mailovou adresu, kterou jsme zadali na stránkách dot48. Měli byste v ní v podstatě okamžitě nalézt e-mail v následujícím tvaru:

Hello[redacted]

Thank you for giving dot42 a try!

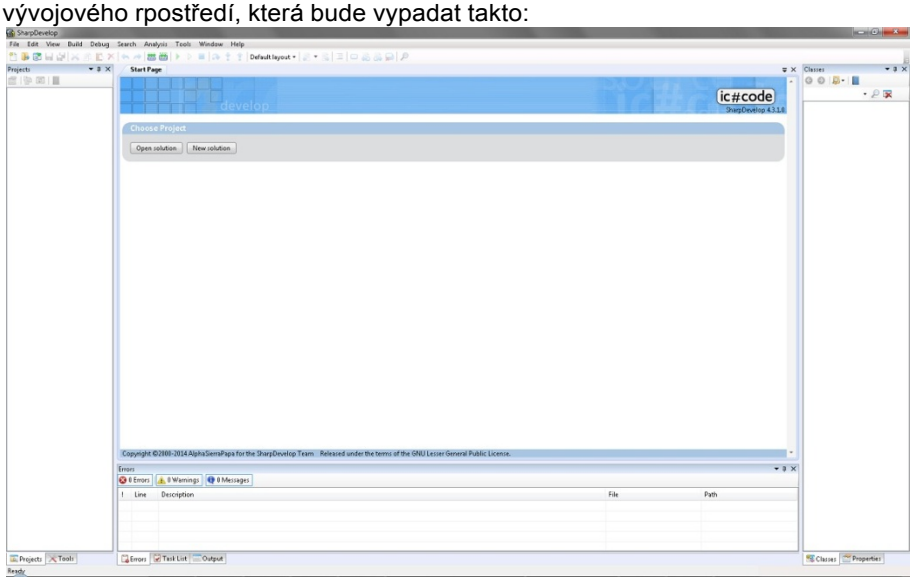
Use this serial number when you run dot42 for the first time.

9a83ee01550307b5

Thank you,
The dot42 Team

Pokud jste e-mail neobdrželi, chvíli počkejte a v případě, že do 10 minut nepřijde, prohlédněte si složku spam ve svém e-mailovém klientu. Jelikož je tento e-mail automaticky generován, může se stát, že ho bude kontrola považovat za spam. Jakmile pak zadáte seriový klíč, instalace by se měla zdárně dokončit.

Jako poslední krok je tedy potřeba nalézt v počítači aplikaci "Sharp Develop (dot42 C# for Android)", kterou vám instalační balíček sám doinstaloval. Pokud ji najdete, zkuste ji spustit a měli byste být přivítáni úvodní obrazovkou vývojového rpostředí, která bude vypadat takto:



I když vám to nejspíše instalátor doporučí, po instalaci není třeba restartovat počítač.

Jakmile jsme splnili všechny kroky, nic nám už nebrání v tom, začít vytvářet C# aplikace pro Android, což si vyzkoušíme v další aktivitě.

7 Ahoj Světe! - naše první C# aplikace na Androidu

Téma	Ahoj Světe! - naše první C# aplikace na Androidu	
Tematický celek	Tvorba mobilní Android aplikace v jazyce C#	
Motivační rámec	Poté, co jsme si nainstalovali vývojová prostředí a nástroje, je čas na naši první C# aplikaci pro Android.	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	PC, tablet nebo emulátor OS Android. Vývojové prostředí SharpDevelop z balíčku dot42.	
Stručný popis aktivity s využitím přístroje	Studenti spustí vývojové prostředí SharpDevelop. Vytoří si autentifikační certifikát pro Android aplikace a vytvoří pomocí něj první aplikaci Hello world. Tu následně importují do tabletu či Android emulátoru.	
Vhodné místo	Běžná počítačová učebna vybavená operačním systémem Windows.	
Cíle aktivity	Studenti budou schopni vytvořit Android aplikaci psanou v jazyce C# a importovat ji do přístroje nebo emulátoru.	
Rozvíjené	Kompetence k učení, k řešení problémů	
Předchozí znalosti	Aktivita navazuje na Instalace a nastavení vývojářských nástrojů dot42	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Spuštění a vytvoření podepisovacího certifikátu.	Samostatná práce nebo práce ve skupině
15 minut	Tvorba aplikace Hello World.	Samostatná práce nebo práce ve skupině
15 minut	Import aplikace do tabletu či emulátoru a její testování.	Samostatná práce nebo práce ve skupině, skupinová diskuse
Hodnocení	Bez hodnocení. Studenti budou hodnoceni až po tvorbě vlastních aplikací.	
Návaznosti	Aktivita C# aplikace pro Android s grafickými prvky	

Zadání:

1. Spusťte vývojové prostředí SharpDevelop.

2. Vytvořte nový projekt typu dot42.

3. Vytvořte si autentifikační certifikát pro svoje Android aplikace.

4. Zkompilujte systémem předvytvořenou aplikaci Ahoj světe.

5. Vytvořenou aplikaci nahrajte do tabletu nebo do Android emulátoru a spusťte.

Postup:

Po spuštění prostředí SharpDevelop je nejprve potřeba vytvořit nový projekt (New solution). Jakmile to uděláme, vyskočí nám nabídka všech možných typů projektů. Nás zajímá ten, který je v sekci C#/dot42 a jmenuje se "dot42Application Project". Dále pak zvolíme název a cestu, kam se má projekt uložit.

Jakmile dáme vytvořit, otevře se nám menu s nastavením projektu. Toto menu je velmi důležité, protože v něm vybíráme identifikační certifikát pro naši aplikaci a také verzi OS Android, pro kterou je aplikace určena. V případě, že tvoříte aplikaci tohoto typu úplně poprvé, je nutné vytvořit autentifikační certifikát, kterým se bude aplikace systému prokazovat. Jakmile tuto volbu vybereme, spustí se intuitivní certifikační průvodce, který vám vše vysvětlí. Proces tvorby certifikátu je pak podrobně znázorněn na videu níže.

Volba verze OS Android je velmi důležitá, protože když zvolíte příliš novou verzi, na starších systémech nemusí fungovat. Pokud používáte emulátor BlueStacks, jako je vidět ve videu, zjistěte si na této stránce, jakou verzi OS Android vaše verze emuluje. V případě ukázkového videa to byla verze 4.0.3.

Jakmile jsme s nastavováním hotovi, vytvoří se ukázková aplikace HelloWorld, která obsahuje dvě hlavní složky. MainActivity.cs je zdrojový soubor C# a MainLayout.xml je šablona pro to, jak má aplikace vypadat. Pak už stačí jen projekt "spustit" (Zelená šipka Run compiled exe) a otevře se menu pro výběr zařízení, do kterého se má aplikace importovat. Zařízení musí být přímo připojeno k počítači buď kabelem, nebo přes Wi-fi síť.

V případě, že používáte emulátor tak, jako je to naznačeno na videu, musíte si zkompilovaný soubor apk najít ve složce projektu a do emulátoru ho doinstalovat ručně. Pak už stačí jen emulátor spustit a aplikaci hned uvidíte mezi nainstalovanými, viz video.

Videoukázka postupu viz. on-line kurz

8 C# aplikace pro Android s grafickými interaktivními prvky

Téma	C# aplikace pro Android s grafickými interaktivními prvky	
Tematický celek	Tvorba mobilní Android aplikace v jazyce C#	
Motivační rámec	Teď když už umíme vytvořit a spustit jednoduchou aplikaci, je třeba posunout k ovládacím grafickým prvkům.	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	PC, tablet nebo emulátor OS Android. Vývojové prostředí SharpDevelop z balíčku dot42.	
Stručný popis aktivity s využitím	Studenti pomocí vývojového prostředí sharpDevelop vytvoří aplikaci využívající dvě tlačítka, která budou plnit různé akce.	
Vhodné místo	Běžná počítačová učebna vybavená operačním systémem Windows.	
Cíle aktivity	Studenti budou schopni vytvořit Android aplikaci, která bude obsahovat ovládací grafické interaktivní prvky, jako jsou např. tlačítka.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů	
Předchozí znalosti	Aktivita navazuje na aktivitu Ahoj Světe! - naše první C# aplikace na Androidu	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Spuštění počítače, založení nového projektu a odstranění přebytečných prvků.	Samostatná práce nebo práce ve skupině
15 minut	Tvorba prvního tlačítka se změnou jeho textu při stisku.	Samostatná práce nebo práce ve skupině
20 minut	Tvorba druhého tlačítka, které zobrazí libovolnou hlášku v novém dialogu a následné vyzkoušení finální aplikace.	Samostatná práce nebo práce ve skupině
Hodnocení	Bez hodnocení. Studenti budou hodnoceni až po tvorbě vlastních aplikací.	
Návaznosti	Na tuto aktivitu navazuje C# aplikace pro Android a práce s animacemi.	

Zadání:

1. Vytvořte nový projekt typu dot42.

2. Vymažte z projektu všechny přebytečné prvky a pojmenujte výslednou aplikaci.

3. Vložte do návrhu stránky nové tlačítko roztažené přes celou obrazovku a při jeho stisku mu změňte jeho text.

4. Vložte do návrhu stránky nové tlačítko, které bude malé a libovolně napozicované pod tlačítkem předchozím. Při stisku na toto tlačítko se objeví informační okno s libovolným textem.

5. Aplikaci nahrajte do zařízení či emulátoru a vyzkoušejte.

Postup:

Nejprve je třeba založit novou aplikaci typu dot42 pro Android v prostředí SharpDevelop. Postup je obdobný jako v předchozí kapitole s tím rozdílem, že tentokrát smažeme textové pole pro hlášku "Ahoj světe", kterou nám SharpDevelop v základu vytváří. toho docílíme tak, že si otevřeme xml soubor MainLayout, který je součástí projektu. Tento soubor je pro práci s grafikou velice důležitý, jak se dozvíme v další části. Postup je pak na následujícím videu: (viz. on-line kurz).

Dalším krokem pak bude samotné vytvoření nového tlačítka. Nejprve však svou aplikaci pojmenujeme. V příkladu jsem použil název "Interaktivní prvky", a jak je vidět na následujícím videu, umístil jsem ho do tagu Label nad hlavní aplikaci pomocí syntaxe [Activity(Label = "Interaktivní prvky")]. To způsobí, že naše aplikace bude mít mezi aplikacemi v našem zařízení patřičný název a nadpis. Dalším krokem je pak vložení samotného tlačítka.

Dot42 řeší grafický design Android aplikací úplně stejně jako všechny ostatní jazyky určené pro tento operační systém. Základní rozvržení grafiky se řeší přes konfigurační xml soubory, tzv. Layouty. Tyto xml soubory mají svoji vlastní syntaxi a možnosti, které jsou velmi podobné CSS (Kaskádovým stylům užívaným na webu). Pokud se programátor chce naučit s grafikou na androidu pracovat, je pro něj naprosto nezbytné se s těmito xml layouty seznámit. Manuály jsou pak k nahlédnutí např. zde:
http://developer.android.com/guide/topics/ui/declaring-layout.html

V příkladě bylo použito tlačítko s následujícím zápisem:

```
<Button
  android:id="@+id/Tlacitko1"
  android:layout_width="fill_parent
  "
  android:layout_height="wrap_content"
  android:text="Zmáčkni mě" />
```

Vysvětlení tohoto zápisu je následující. Vlastnost **android:id="@+id/** registruje tlačítko v aplikaci pod určitým

názvem, pomocí něhož se bude s tlačítkem pracovat. V našem případě jsme použili jméno Tlacitko1. Další použitá vlastnost byla **android:layout_width**, jehož hodnotu jsme nastavili na **fill_parent.** Tato vlastnost udává šířku tlačítka a naše zvolená hodnota pak určuje, že tlačítko bude roztaženo přes celou obrazovku. Podobně je na tom pak vlastnost **android:layout_height**, která udává výšku. V obou těchto případech jsme mohli nastavit výšku v bodech, ale místo toho jsme použili předdefinované možnosti Android layoutu. V případě výšky to bylo **wrap_content**, což znamená, že se tlačítko roztahuje podle toho, co je v něm napsáno. Poslední byla vlastnost **android:text**, která jak jistě každému dojde, určuje, co bude na tlačítku napsáno.

Jakmile jsme měli grafickou část tlačítka, bylo třeba vytvořit funkcionalitu. V hlavním programu jsme

```
proto přidali: var tlacitko = (Button)FindViewById(R.Ids.Tlacitko1);
tlacitko.Click += (x,s) => tlacitko.Text = "Zmáčknul jsi mě";
```

Tyto dva řádky jsou pak samy o sobě vcelku pochopitelné. Na prvním využíváme funkci **FindViewById**, která vrátí odkaz na specifikovaný objekt, který se nachází na layoutu, tedy v grafice aplikace. Funkce ho hledá podle id předtím specifikovaného ve
vlastnosti **android:id="@+id/.** Funkce hledá univerzálně objekty, proto je třeba výsledek přetypovat na Button. Na další řádce pak přidáme mezi listenery buttonu na událost Klik zkrácený zápis event handleru, který bere vždy parametry (object sender, eventArgs parametry), zde definované jako (x, s). Pokud máme událost, která dělá jen jednu akci na jednom řádku, vyplatí se tento zápis psát zkráceně, jak je tomu v příkladu. U dalšího tlačítka použijeme zápis klasicky celý, který je běžným uživatelům jazyka C# už povědomější. Celou tuto část pak ilustruje následující video (viz. on-line kurz).

Naším dalším úkolem bylo vytvořit tlačítko, po jehož kliknutí na nás vyskočí námi definovaná hláška. Tentokrát jsme v layoutu použili trochu jiné parametry, a to následující:

```
<Button
 android:id="@+id/Tlacitko2"
 android:layout_width="150dp"
 android:layout_marginLeft="15dp"
 android:layout_marginTop="80dp
 "
 android:layout_height="wrap_content"
 android:text="Ukaž mi zprávu"/>
```

Je tedy třeba probrat několik změn a novinek. Do **android:layout_width** jsme tentokrát specifikovali šířku v bodech a přidali jsme dvě nové vlastnosti, a to **android:layout_marginLeft** a **android:layout_marginTop**. Obě tyto vlastnosti řeší velikost mezer kolem našeho tlačítka. Vlastnost **marginLeft** z leva a vlastnost **marginTop** ze shora. U obou jsme nastavili příslušný počet bodů. Výsledek tohoto rozvržení si pak můžeme prohlédnout v ukázce finální aplikace.

Když jsme tedy hotovi i s druhým tlačítkem, je třeba přidat jeho funkcionalitu. Podobně jako u prvního tlačítka mu musíme přidat funkci reagující na událost stisknutí, takzvaný handler. Tentokrát však použijeme klasický zápis a vytvoříme si na to celou novou funkci. Začátek je velmi podobný:

```
var tlacitko2 =
 (Button)FindViewById(R.Ids.Tlacitko2);
 tlacitko2.Click += ZobrazZpravu;
```

A teď musíme vytvořit funkci ZobrazZpravu. Její zápis pak vypadá následovně: private void ZobrazZpravu(object zdroj,

```
EventArgs parametry){
  var tvurce = new AlertDialog.Builder(this);
  tvurce.SetMessage("Toto je naše ukázková zpáva!"); var okno = tvurce.Create();
  okno.Show();
}
```

Jak jsem již psal předtím, klasická funkce reagující na událost má tyto základní parametry (object zdroj, EventArgs parametry). Když takovou funkci vyrobíme, můžeme jí přiřadit jako handler události. V našem případě má sloužit k zobrazení dialogové hlášky. Proto je třeba nejprve si vyvrořit **AlertDialog Builder** a uložit si ho do proměnné. Jakmile ho máme, musíme mu přesně specifikovat, jakou zprávu chceme zobrazit.
Na to nám poslouží funkce **SetMessage**. Jakmile máme takto builder nastaven, stačí mu jen přikázat, aby vyrobil instanci okna pomocí funkce **Create** a následně ho ukázat pomocí funkce **Show**.

Celý postup je pak vidět na následujícím videu (viz. on-line kurz).

A tím pádem je celá aplikace hotová a je třeba ji exportovat do emulátoru nebo do libovolného zařízení s OS Android. Na následujícím videu můžeme vidět, jak vypadá a jak se chová (viz. on-line kurz).

9 C# aplikace pro Android a práce s animacemi

Téma	C# aplikace pro Android a práce s animacemi	
Tematický celek	Tvorba mobilní Android aplikace v jazyce C#	
Motivační rámec	Když už jsme si pohráli s klasickými ovládacími prvky, vyzkoušíme si teď vytvořit jednoduchou animaci. (Aneb je to barevné a hýbá se to!)	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	PC, tablet nebo emulátor OS Android. Vývojové prostředí SharpDevelop z balíčku dot42.	
Stručný popis aktivity s využitím přístroje	Studenti pomocí vývojového prostředí sharpDevelop vytvoří aplikaci, generující animaci pohybujícího se kruhu.	
Vhodné místo	Běžná počítačová učebna vybavená operačním systémem Windows.	
Cíle aktivity	Studenti budou schopni vytvořit Android aplikaci, která bude obsahovat jednoduchou animaci libovolného geometrického tvaru, například kruhu.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů	
Předchozí znalosti	C# aplikace pro Android s grafickými interaktivními prvky	
Časový plán	Fáze činnosti s přístrojem	Metody a formy,
10 minut	Spuštění počítače, založení nového projektu a odstranění přebytečných prvků.	Samostatná práce nebo práce ve skupině
35 minut	Tvorba nového panelu, na kterém se bude pohybovat zelený kruh s nápisem uvnitř.	Samostatná práce nebo práce ve skupině
Hodnocení	Bez hodnocení. Studenti budou hodnoceni až po tvorbě vlastních aplikací.	
Návaznosti	Na tuto aktivitu navazuje aktivita Závěrečné poznámky a zdroje	

Zadání:

1. Vytvořte nový projekt typu dot42.

2. Vyčistěte ho od zbytečností, včetně MainLayout.xml, tentokrát nebude potřeba.

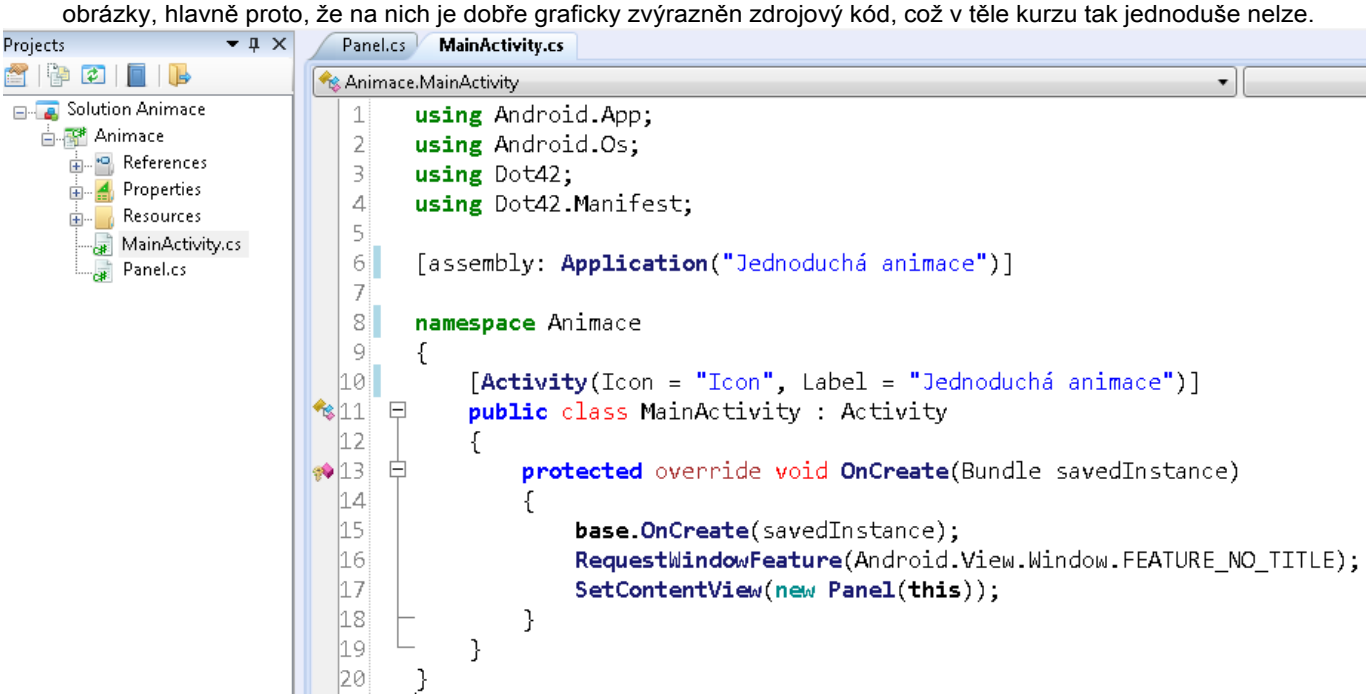
3. Nastavte aplikaci na celou obrazovku bez nadpisu.

4. Vytvořte panel s modrým pozadím, na kterém bude probíhat animace pohybujícího se zeleného kruhu s nápisem uvnitř.

5. Aplikaci spusťte a vyzkoušejte.

Postup:

Nejprve je třeba vytvořit nový projekt a vyčistit ho od zbytečností, což už jsme dělali několikrát. Bylo by také dobré nastavit aplikaci jméno, jako jsme si ukázali v předchozí kapitole. Co však tentokrát uděláme jinak je, že odstraníme také MainLayout.xml, který nám normálně umožňuje stavět aplikaci pomocí interaktivních grafických komponent. Na animaci a vykreslování elementární grafiky však nestačí, na to budeme potřebovat nějaký panel, který bude potomkem třídy **SurfaceView**. To bychom ale předbíhali, nejprve je třeba upravit hlavní vygenerovaný soubor a to je **MainActivity.cs**. Na následujících obrázcích bude ilustrováno, jak na to. Pro demonstraci byly zvoleny obrázky, hlavně proto, že na nich je dobře graficky zvýrazněn zdrojový kód, což v těle kurzu tak jednoduše nelze.



Jak vidíme na obrázku, zas tolik se nám to nezměnilo. Přibyla nám funkce **RequestWindowFeature**, která zajišťuje to, aby se aplikace spustila v určitém náhledovém režimu. V našem případě chceme aplikaci na full screen bez nadpisu, jako to můžeme vidět v předchozí aktivitě. Vzhledem k tomu, že chceme mít modrou plochu na celé obrazovce, nadpis s názvem aplikace by nám tam zbytečně překážel a ubíral prostor. Poté použijeme naši již známou funkci **SetContentView**, abychom nastavili určitou grafiku jako hlavní obsah aplikace. Tentokrát však nepoužijeme layout vytvořený v MainLayout.xml, ale námi vytvořený nový panel, který je potomkem třídy **SurfaceView**. Na následujících obrázcích si pak ukážeme funkcionalitu, kterou ho naplníme.

```
using System.Threading;
using Android.Content;
using Android.Graphics;
using Android.View;
using Java.Lang;

namespace Animace
{
    internal class Panel : SurfaceView, ISurfaceHolder_ICallback, IRunnable
    {
        private int poloha = 50; //poloha kruhu
        private int rozdil = 1; //posun doleva či doprava
        private bool bezi; //je spuštěná aplikace?
        private readonly ISurfaceHolder holder; //odkaz na samotný panel a jeho grafiku

        public Panel(Context obsah) : base(obsah)
        {
            holder = GetHolder();
            holder.AddCallback(this);
        }
```

Na tomto obrázku je vidět definice samotné třídy **Panel**. Jak vidíme, je potomkem třídy **SurfaceView**, která je specificky navržena, aby poskytla aplikaci co nejlepší možnosti v oblasti grafiky a kreslení na "plátno" neboli canvas. Třída **Panel** pak zároveň implementuje dvě rozhraní. **ISurfaceHolder_ICallback** je rozhraní, které umožňuje programu přijímat informace o změnách na "povrchu" aplikace.
IRunnable je pak klasické rozhraní, které umožňuje třídě běžet v samostatném vlákně.

Dále pak nadefinujeme několik uživatelských proměnných. Proměnné **poloha** a **rozdil** se budou týkat polohy a směru pohybu našeho kruhu a proměnná **bezi** bude rozhodovat o tom, zda se bude ve vlákně přepočítávat poloha, a bude hlídat, zda aplikace už neskončila. Poslední proměnná **holder** bude v sobě udržovat odkaz na grafický "povrch" aplikace, na kterém se bude vykreslovat naše grafika.

A jako poslední v této fázi je třeba nadefinovat konstruktor **Panelu**, ve kterém je třeba získat aktuální holder (pomocí funkce **GetHolder**) a spojit ho s touto instancí třídy Panel funkcí **AddCallback**. To zaručí, že změny vycházející z této instance panelu se do grafiky skutečně promítnou.

Dále je třeba vytvořit funkce vyžádované díky implementaci našich dvou rozhraní.

```
public void SurfaceCreated(ISurfaceHolder iSurfaceHolder)
{
    var thread = new Thread(this);
    bezi = true;
    thread.Start();
}

public void SurfaceChanged(ISurfaceHolder iSurfaceHolder, int int32, int int321, int int322)
{
}

public void SurfaceDestroyed(ISurfaceHolder iSurfaceHolder)
{
    bezi = false;
}

public void Run()
{
    while (bezi)
    {
        var canvas = holder.LockCanvas();
        if (canvas != null)
        {
            DoDraw(canvas);
            holder.UnlockCanvasAndPost(canvas);
            Thread.Sleep(5);
        }
    }
}
```

Všechny tyto funkce začínající slovem surface jsou vyžadovány rozhraním **ISurfaceHolder_ICallback** a všechny se týkají samotného povrchu aplikace. **SurfaceCreated** se zavolá v okamžiku vytvoření grafického povrchu. V tu chvíli už můžeme nastavit proměnnou **bezi** na true, protože je načase začít vykreslovat animaci. Zároveň pak vytvoříme nové vlákno, ve kterém se bude animace připravovat a vykreslovat a toto vlákno nastartujeme. Mechanika je stejná jako v jakékoliv jiné C# aplikaci. Ve funkci **SurfaceDetroyed** vykreslování zase naopak vypneme, protože aplikace byla zavřena a povrch smazán. Funkce **SurfaceChanged** si nevšímáme, protože pro ni v naší aplikaci nemáme využití. Nicméně v programu se musí nacházet, protože její užití je vynuceno rozhraním.

Poslední funkce vynucená rozhraním je pro nás klíčová. Jedná se o funkci **Run**, kterou využívá rozhraní **IRunnable**. Tato funkce se spouští, hned jak je nastartováno vlákno, ve kterém běží, a jakmile doběhne, vlákno se ukončí. Ve většině případů se v něm nachází nějaký cyklus, který dokola opakuje určité akce. V našem případě opakujeme cyklus, dokud aplikace běží. Nejprve pomocí funkce **LockCanvas** zamkneme plátno pro jiné procesy a systému tím dáme najevo, že se chystáme měnit samotné pixely na něm, jinak řečeno, budeme na něj něco kreslit. Poté zavoláme námi vytvořenou funkci **DoDraw** (probereme ji dále), která se postará o samotné vykreslení grafiky na plátno. Pak už jen stačí pomocí funkce **UnlockCanvasAndPost** plátno odemknout a zobrazit na něm naše změny. Dál už stačí jen naše vlákno na chvíli uspat pomocí **Thread.Sleep**, abychom simulovali plynulý pohyb kruhu, a funkcionalita je na světě. Teď už nám stačí jen dopsat funkci **DoDraw**, která se postará o samotné umístění grafických prvků.

```
private void DoDraw(Canvas canvas)
{
    // posun kruhu po ploše
    poloha += rozdil;
    if (poloha > 250)
    {
        rozdil = -1;
    }
    else if (poloha < 30)
    {
        rozdil = 1;
    }

    // Nastavíme pozadí na modrou barvu
    canvas.DrawColor(Color.BLUE);
    var paint = new Paint();
    paint.SetTextAlign(Paint.Align.CENTER);

    // Vykreslíme zelený kruh
    paint.SetColor(Color.GREEN);
    canvas.DrawCircle(poloha, 140.0f - poloha / 3, 45.0f, paint); //souřadnice řeší pohyb

    //A červený nápis doprostřed kruhu
    paint.SetColor(Color.RED);
    canvas.DrawText("Dot42", poloha, 140.0f - poloha / 3, paint);

}
```

V první části je jednoduchá mechanika, která řeší, zda se kruh bude zrovna pohybovat směrem doleva nebo doprava. Jakmile je poloha rozhodnuta, přichází samotné vykreslování grafiky. Nejprve zabarvíme celé plátno modrou barvou pomocí **DrawColor**(Color.BLUE). Poté vytvoříme nový štětec typu **Paint**, kterým budeme jednotlivé prvky kreslit. Pomocí **SetTextAlign**(Paint.Align.CENTER) nastavíme zarovnávání textu psaného naším štětcem na střed, což použijeme později při vpisování textu do našeho kruhu. Dříve než něco vykreslíme, vždy nastavíme barvu daného objektu pomocí funkce **SetColor**. V případě kruhu to bude zelená a v případě textu pak červená. Na samotné objekty pak použijeme funkce **DrawCircle** v případě kruhu a **DrawText** v případě textu. K umístění pak použijeme souřadnice vygenerované s pomocí naší proměnné **poloha**. A jako jeden z parametrů funkcí také přidáme námi vygenerovaný štětec. Takto zapsaný kód je vcelku jednoznačný a intuitivní. Jakmile jsme tedy dopsali funkci **DoDraw**, nic už nám nebrání aplikaci spustit a podívat se na výsledek.

A na následujícím videu je pak už vidět samotná hotová aplikace. Omluvte prosím cukání v pohybu kruhu, je to způsobeno nahráváním videa z obrazovky, které nedokáže snímat tak rychle, aby byl pohyb plynulý. V samotné aplikaci se kruh pohybuje velmi plynule. (viz. on-line kurz)

10 Závěrečné poznámky a zdroje

Slovo na závěr

Po prostudování tohoto kurzu by měl být student alespoň zběžně vpraven do problematiky vývoje C# aplikací pro

OS Android. U této problematiky je v první řadě klíčové uvědomit si, proč by měl programátor k vývoji Android aplikací vůbec používat C#. Na tuto otázku neexistuje jednoznačná odpověď a sám autor kurzu není docela přesvědčen o tom, že je to ideální cesta. Primárním jazykem pro vývoj takových aplikací je Java a ta je jazyku C# velice podobná a nedalo by se říct, že je v nějakém ohledu slabší. Dokonce existuje programátorský proud, který ji nad C# značně vyzdvihuje kvůli její multiplatformovosti. Na druhou stranu existují programátoři, kteří C# preferují zase z jiných důvodů. Autorovi například připadá jako jazyk uživatelsky více přívětivý a snazší k učení, nicméně to je věc individuálního názoru.

Pokud si tedy po průchodu tímto kurzem nejste úplně jisti, jestli se C# na Androidu věnovat naplno, zvažte, jaký jste typ programátora, vaše okolnosti a předchozí zkušenosti. Pokud jste se ve své profesionální kariéře věnovali převážně technologiím .NET a nechcete na tom nic měnit, odpověď jasně směřuje k užívání C# i k tvorbě vašich mobilních aplikací. Pokud však jazyk C# v sobě nemáte tak hluboce zakořeněn a hledáte pro svou budoucí kariéru ideální způsob tvorby mobilních aplikací, bude pro vás lepší přesunout se do hlavního proudu tvorby mobilních aplikací na OS Android a přeučit se na Javu. Jazyky jsou si syntaxí a možnostmi velmi podobné a zkušenému programátorovi tento přechod nebude činit potíže.

Zdroje:

V tomto kurzu byl použit následující multimediální obsah:

Obrázky:

Kapitola 2:

Obrázek Android robota se znakem C# (obálka volně dostupné elektronické knihy Mono for Android - Create amazing Android apps with C# and .NET, zdroj přímo v kapitole)

Kapitola 3:

Ilustrační obrázek PC (pixabay.com, volné dílo) Ilustrační obrázek tabletu (pixabay.com, volné dílo) Logo dot42

(wikipedia.org , volné dílo)

Logo Xamarin (wikipedia.org , volné dílo) Logo BlueStacks (wikipedia.org , volné dílo)

Kapitola 4:

Logo .NET (wikipedia.org , volné dílo) Logo Android (wikipedia.org , volné dílo)

Kapitola 5:

Logo Xamarin (wikipedia.org , volné dílo) Logo dot42 (wikipedia.org , volné dílo)

Ukázka zdrojového kódu Xamarin (nasnímáno autorem kurzu) Ukázka zdrojového kódu dot42 (nasnímáno autorem

kurzu)

Kapitola 6:

Několik obrázků s postupem instalace nástrojů dot42 (nasnímáno autorem kurzu)

Kapitola 9:

Několik obrázků se zdrojovým kódem aplikace s animací (nasnímáno autorem kurzu)

Videa:

Kapitoly 7,8 a 9:

Videa demonstrující postup práce v daných kapitolách a ukázky finálních produktů. Všechny byly zaznamenány autorem pomocí volně šiřitelného software a umístěny na server youtube pouze pro účely tohoto kurzu.

TVORBA MOBILNÍ ANDROID APLIKACE ZALOŽENÉ NA ANIMACI A DALŠÍCH TECHNIKÁCH

1 Základní informace o projektu

Název

Tvorba mobilní Android aplikace založené na animaci

Anotace programu/zaměření/hlavní cíl

Cílem projektu je postupně zvládnout tvorbu mobilní Android aplikace s využitím programů založených na animaci, jako je např. Adobe Flash Professional nebo Stencyl.

Cílová skupina

Žáci 1. až 3. ročníků středních škol a odpovídajících ročníků gymnázií.

Pomůcky

Osobní počítač (popř. notebook); Adobe Flash Professional s podporou jazyka ActionScript 3.0 (tzn. Adobe Flash CS3 a vyšší); aplikace Stencyl; tablet nebo mobilní telefon s operačním systémem Android (případně emulátor pro OS Android); digitální fotoaparát.

Vazba na RVP

- RVP pro Gymnázia: Vzdělávací oblast Informatika a informační a komunikační technologie,
- RVP pro střední odborné vzdělávání se vzdělávací oblastí: Vzdělávání v informačních a komunikačních technologiích (jejichž koncepce ŠVP zahrnuje tematické vyučovací celky orientované na rastrovou, vektorovou grafiku a základy algoritmizace.

Mezipředmětové vazby

V závislosti na jednotlivých aktivitách.

Fáze projektu

1. Seznámení se s možnostmi realizace Android aplikací bez nutnosti hluboké znalosti programovacího jazyka.
2. Příprava prostředí Adobe Flash Professional a instalace platformy Adobe AIR pro vývoj Android aplikací a jejich spouštění.
3. Tvorba Android aplikací a jejich následné publikování do zařízení s OS Android.
4. Prezentace vlastních aplikací.
5. Hodnocení.

2 Motivační rámec projektu

Odkud se berou aplikace pro tablety a mobilní telefony?

Jistě už některé z vás napadla myšlenka, že by bylo skvělé vytvořit si vlastní aplikaci pro tablet nebo dotykový telefon s operačním systémem Android. Také jste možná přemýšleli o tom, odkud se bere to nepřeberné množství všech těch aplikací a her např. na Google play. Možná tomu nebudete věřit, ale hry a aplikace nevyrábí jen společnosti jako Gameloft nebo Glu, ale také jednotlivci nebo malé několikačlenné skupiny, jako třeba vy a vaši kamarádi (spolužáci apod.), tak proč se nepokusit realizovat vlastní aplikaci pro mobilní zařízení?

Android aplikace bez nutnosti rozsáhlých znalostí programovacích jazyků?

Někteří z vás si jistě řeknou, že bez znalosti programovacího jazyka, jakým je např. Java, C# nebo Python (a dalších rozšiřujících balíčků k těmto jazykům) snad ani není možné mobilní aplikaci vytvořit, ale opak je pravdou. Abyste vytvořili pěknou uživatelsky přívětivou aplikaci nebo hru, nemusíte v současnosti znát ani jeden z výše uvedených jazyků, protože existují nástroje, které vám mohou vývoj vaší aplikace usnadnit a váš sen o vlastní mobilní aplikaci uskutečnit.

Pokud patříte k těm, kteří vždy chtěli vytvořit aplikaci pro mobilní zařízení (s OS Android) a učit se programovat až potom, pak právě pro Vás je určen tento projekt. A nejen pro vás.

Doporučený multimediální materiál

- Odkazy na 30denní zkušební verzi Adobe Flash Professional CC (odkaz viz. on-line kurz)

video viz. on-line kurz

- Odkaz na aplikaci STENCYL (odkaz viz. on-line kurz)

video viz. on-line kurz

- Odkaz na aplikaci GameSalad Creator (odkaz viz. on-line kurz)

video viz. on-line kurz

3 Poznámky k využití přístrojů a nástrojů

Bez čeho se při vývoji aplikací animačními technikami neobejdeme?

Ani v dnešní době se většina vývojářů nejrůznějších počítačových programů a aplikací neobejde bez svého osobního počítače, notebooku (laptop, 2 v 1) nebo jiného zařízení s dostatečně velkým zobrazovacím panelem (monitorem) a operačním systémem s podporou celé řady prostředí pro vývoj aplikací.

PC

Notebook

Při grafickém návrhu vzhledu aplikace si jistě člověk vystačí pouze s myší, ale vždy pomůže a celou práci ještě více usnadní co největší plocha dotykového zobrazovacího panelu. Bohužel takové zařízení bývá i v dnešní době finančně nákladné, a tak si člověk, který není zvyklý při kreslení používat myš, musí vystačit např. s tabletem a pro přesnější práci spolu s ním může použít stylus. Tablet nebo mobilní telefon s dotykovým displejem a operačním systémem Android zároveň využijeme ke spouštění našich vytvořených aplikací.

Tablet

Vývojářské nástroje

K vývoji aplikací založených na animaci není nutné znát syntaxi konkrétních programovacích jazyků, často stačí jen aplikovat logické postupy v programech, které podporují tento způsob vývoje Android nebo také iOS aplikací. Mezi takové můžeme zařadit např. programy GameSalad nebo Stencyl, které jsou primárně určeny pro vývoj her a jejich nespornou výhodou je především jejich cena (obě jsou volně dostupné ke stažení a užívání). (odkazy viz. on-line kurz)

GameSalad® stencyl

Speciální výjimku tvoří dlouhá léta vyvíjený a zdokonalovaný nástroj Flash Professional od firmy Adobe. Tento nástroj také umožňuje realizovat aplikace založené na technikách animace a zároveň nabízí plnou podporou programovacího jazyka ActionScript s implementací pokročilých objektově orientovaných programovacích technik. Jeho nevýhodou může být cena, která se pohybuje v řádu několika tisíc korun. Společnost Adobe však často nabízí (aktuálně v rámci služby Creative Cloud) i zvýhodněné programové "balíky" pro školy nebo studující jednotlivce. (odkaz viz. on-line kurz)

Aplikace vyvinuté v prostředí Adobe Flash Professional potřebují ke svému spuštění na různých zařízeních platformu Adobe AIR, která vývojářům umožňuje používat osvědčené webové technologie k tvorbě internetových aplikací a implementovat tyto technologie tak, aby se daly snadno používat a uživatel je měl stále při ruce. (odkaz viz. on-line kurz)

Emulátory

V případě, že nevlastníme nebo po dobu vývoje nemáme k dispozici tablet nebo telefon s dotykovým displejem a s operačním systémem Android, přichází v úvahu efektivní řešení v podobě Android emulátoru, kterých existuje celá řada a zvládnou po všech stránkách (snad kromě telefonování ☺) zastoupit fyzické zařízení. Často mají tu výhodu, že podporují speciální funkce pro vývojáře a většina z nich je volně dostupná. Mezi nejznámější emulátory patří např. BlueStacks, YouWave, Jar Of Beans, GenyMotion. (odkazy viz. on-line kurz)

4 Informační tablo naší třídy jako aplikace v mobilu (tabletu)

Téma	Informační tablo naší třídy jako aplikace v mobilu	
Tematický celek	Tvorba mobilní Android aplikace založené na animaci	
Motivační rámec	Myslete na maturitní ples už dnes a prezentujte svou třídu se svými spolužáky prostřednictvím společného maturitního tabla v podobě Android aplikace. Nic pak nebude bránit tomu, aby si např. v půlnočních novinách někdo z vašich blízkých takové tablo stáhl do svého mobilního zařízení vyfocením QR kódu.	
Počet žáků	15 (nebo podle počtu počítačových stanic v učebně)	
Věk žáků	15-18	
Pomůcky	Osobní počítač připojený k internetu, digitální fotoaparát nebo mobilní telefon, tablet, emulátor OS Android, program Adobe Flash Professional CS3 a vyšší; editor vektorové nebo rastrové grafiky.	
Stručný popisaktivity s využitím přístroje	Žáci nafotí svoje spolužáky digitálním fotoaparátem (nebo mobilním telefonem), spustí prostředí Adobe Flash Professional, ve kterém vytvoří interaktivní prezentaci se stručnými informacemi o svých spolužácích, doplněnou o jejich fotografii. Tunásledně publikují do tabletu, mobilního telefonu s OS Android nebo do Android emulátoru s využitím instalace platformyAdobe AIR.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Žáci budou schopni efektivně vytvořit jednoduchou Android aplikaci v podobě interaktivní prezentace a publikují ji na mobilní zařízení; budou ovládat základní principy plošné animace.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů, komunikativní kompetence, kompetence využívat prostředky informačních a komunikačních technologií a práce s informacemi.	
Předchozí znalosti	Aktivita navazuje na tematické oblasti učiva zahrnující realizaci a editaci rastrových a vektorových obrazů.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
25 minut	Fotografování spolužáků digitálním fotoaparátem (popř. mobilním zařízením s fotoaparátem). Získávání a zaznamenávání základních informací o spolužácích (využití aplikací pro psaní textu na mobilním zařízení nebo na osobním počítači).	Samostatná práce se spoluúčastí ostatních spolužáků při fotografování portrétů
45 minut	Realizace interaktivní prezentace - "Maturitní tablo třídy" v prostředí Adobe Flash Professional na osobním počítači.	Samostatná práce podle dostupných materiálů
15 minut	Návrh ikony aplikace a její export v různých velikostech v Adobe Flash Pro (popř. v programu orientovaném na tvorbu a editaci rastrové či vektorové grafiky s možností exportu do rastrového obrazu).	Samostatná práce podle dostupných materiálů
20 minut	Publikování hotové aplikace a import do mobilního zařízení s OS Android nebo do odpovídajícího emulátoru.	Samostatná práce podle dostupných materiálů
20 minut	Ukázky (krátké prezentace) jednotlivých aplikací mezi spolužáky, skupinové hodnocení prací. Diskuse na téma: Jaké další aplikace (prezentace) by mohly být realizovány stejným způsobem?	Práce ve skupině
Hodnocení	Hodnocení celkové funkčnosti a přehlednosti vytvořené aplikace, hodnocení grafického návrhu prostředí aplikace (dodržení kontrastu barva textu versus pozadí aplikace).	
Návaznosti	Na tuto aktivitu může navazovat návrh aplikace s využitím formulářových komponent v prostředí Adobe Flash.	

Doporučený multimediální material

1. Příprava pozadí aplikace, nadpisu a manipulace s pracovní plochou prostředí Adobe Flash Professional:
2. Import fotografií a obrázků do knihovny v Adobe Flash Professional:
3. Vkládání fotografií a obrázků (importovaných do knihovny) na pracovní plochu v Adobe Flash Professional, základní transformace obrázků a fotografií a jejich umístění na ploše:
4. a) Převod obrázku či fotografie na tlačítko, tvorba vlastního obrázkového tlačítka:
   b) Vkládání nových klíčových snímků na časové ose projektu a zvýšení počtu pracovních ploch v připravované aplikaci:
5. Volání události tlačítka po stisknutí s odkazem na definovaný klíčový snímek:
   - přechod na další "obrazovku":
   - přechod na informace o spolužácích:

   Sami zkuste vytvořit nové klíčové snímky, kdy každý z těchto snímků bude sloužit jako jednoduchý profil pro jednotlivé vaše spolužáky (do všech "profilů" nezapomeňte vložit větší fotografii se spolužákem a základní textové informace, které o spolužákovi máte). Z malých fotografií (náhledů na úvodních obrazovkách), které jste si připravili jako symboly tlačítek, vytvořte interaktivní tlačítka s odkazy na jednotlivé profily spolužáků.

6. Publikování finální aplikace a její spuštění v operačním systému Android.

   Před publikováním (exportováním a spuštěním aplikace na vašem mobilním zařízení nebo v emulátoru) je nutné vytvořit 3 vzhledově stejné rastrové ikony, které budou vaši aplikaci reprezentovat na příslušném zařízení. Fantazii při návrhu ikon se meze nekladou, je však nutné respektovat několik pravidel, bez kterých by výslednou aplikaci nebylo možné publikovat:

   - rastrové obrázky ikon musí být ve formátu PNG (výhodou je podpora průhlednosti);
   - rastrové obrázky ikon musí být ukládány (exportovány) ve velikostech "opsaného čtverce" (v pixelech): 36 x 36, 48 x 48, 72 x 72.

   Podobné aplikace v podobě interaktivní prezentace mohou být koncipovány k nejrůznějším účelům, nejen jako tablo prezentující žáky, ale podobně může být princip takové aplikace využit při tvorbě sbírky vzorečků nebo krátkých poznámek k nejrůznějším předmětům či jako prezentace školy, sborník nejrůznějších oblíbených fotografií, restaurací, jídel, nápojů, receptů nebo jiných postupů. Se spolužáky zkuste vymyslet i další koncepční využití takové aplikace.

všechna videa viz. on-line kurz

5 Tvoříme hru pro Android

Téma	Vyrábíme hru pro Android	
Tematický celek	Tvorba mobilní Android aplikace založené na animaci	
Motivační rámec	Hry nepochybně patří mezi nejpopulárnějších formy mobilních aplikací. Existuje mnoho kategorií her a nejčastěji se hry dělí podle žánru (např. akční, logické, arkádové, strategické, RPG, simulátory (sportovní, závodní, letecké atd.). Pojďme si společně jednu takovou jednoduchou hru společně vytvořit.	
Počet žáků	15 (nebo podle počtu počítačových stanic v učebně)	
Věk žáků	15-18	
Pomůcky	Osobní počítač připojený k internetu; mobilní telefon s dotykovým displejem nebo tablet s OS Andoid popř. emulátor OS Android; program Stencyl; editor vektorové nebo rastrové grafiky.	
Stručný popis aktivity s využitím přístroje	Žáci v editoru rastrové grafiky nebo v editoru vektorové grafiky (s následným exportem do rastrového obrazu) vytvoří obrázky herních postav (hrdinů a padouchů), bonusových objektů, které budou v průběhu hry sbírány, a obrázky herního prostředí (pozadí hry, bloky "země", po kterých se bude hráč pohybovat). Ve vývojovém prostředí programu Stencyl žáci vytvořené obrázky použijí při realizaci vlastní hry. Výslednou hru žáci otestují na svém zařízení s operačním systémem Android (mobilní telefon, tablet) nebo v Android emulátoru.	
Vhodné místo	Počítačová učebna	
Cíle aktivity	Žáci budou schopni efektivně vytvořit jednoduchou Android hru a publikovat ji na mobilní zařízení; budou ovládat základní principy a postupy při návrhu her pro Android.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů, kompetence využívat prostředky informačních a komunikačních technologií a práce s informacemi.	
Předchozí znalosti	Aktivita navazuje na tematické oblasti učiva zahrnující realizaci a editaci rastrových a vektorových obrazů.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 až 60 minut	Příprava vlastních grafických objektů do hry na stolním PC nebo notebooku v libovolném grafickém editoru rastrové grafiky (např. Gimp, Photoshop, Pixlr) nebo editoru vektorové grafiky (např. Inkscape, Adobe Illustrator, CorelDRAW) s exportem obrazového materiálu do rastrové grafiky.	Samostatná práce
15 minut	Návrh scénáře jednoduché herní zápletky.	Samostatná práce s diskuzí
60 až 90 minut	Realizace jednoduché plošinové hry ve vývojovém prostředí Stencyl.	Samostatná práce podle dostupných materiálů
20 minut	Publikování hotové aplikace a import do mobilního zařízení s OS Android nebo do odpovídajícího emulátoru.	Samostatná práce podle dostupných materiálů
30 minut	Ukázky (krátké prezentace) jednotlivých aplikací mezi spolužáky, skupinové hodnocení prací. Diskuse na téma: Jak bych mohl svoji hru vylepšit nebo co bych měl ve hře změnit?	Skupinová diskuze
Hodnocení	Hodnocení celkové funkčnosti a přehlednosti vytvořené aplikace, hodnocení grafického návrhu prostředí aplikace (dodržení kontrastu barvy prostředí a herních postav versus pozadí aplikace).	
Návaznosti	Na tuto aktivitu může navazovat společná skupinová realizace rozsáhlejší hry a její publikace na oficiálním serveru s Android aplikacemi.	

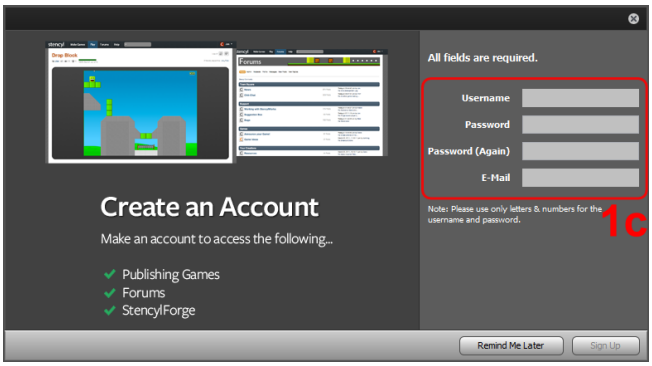
Doporučený multimediální materiál

1. Poslední verzi programu Stencyl je možné stáhnout přímo z webových stránek produktu
2. na www.stencyl.com/download/. Startovací verze programu (Starter) je zdarma, avšak placené verze (Indie a Studio), jak už to tak bývá, nabízejí další rozšíření. Nejdražší verze Studio pak zahrnuje vše, co verze Indie, další podporu uvedenou níže v tabulce.

Indie	Studio
Čerstvý přístup k programovým novinkám	Přímé publikování na iOS a Android
Publikování do platforem Flash, Windows, Mac a Linux	Podpora reklam typu: iAd, AdMob, a další
Žádné vkládání vodoznaků nebo vynucených značkovacích prvků	Podpora Game Center pro iOS
Přístup do zákaznického fóra	Podpora nákupů pro iOS a Android v aplikacích
Vkládání vlastních reklam a vlastních načítacích obrázků pro Flash před hraním her	Zásuvné moduly třetích stran s dalšími funkcemi

3. Instalace a spuštění programu Stencyl

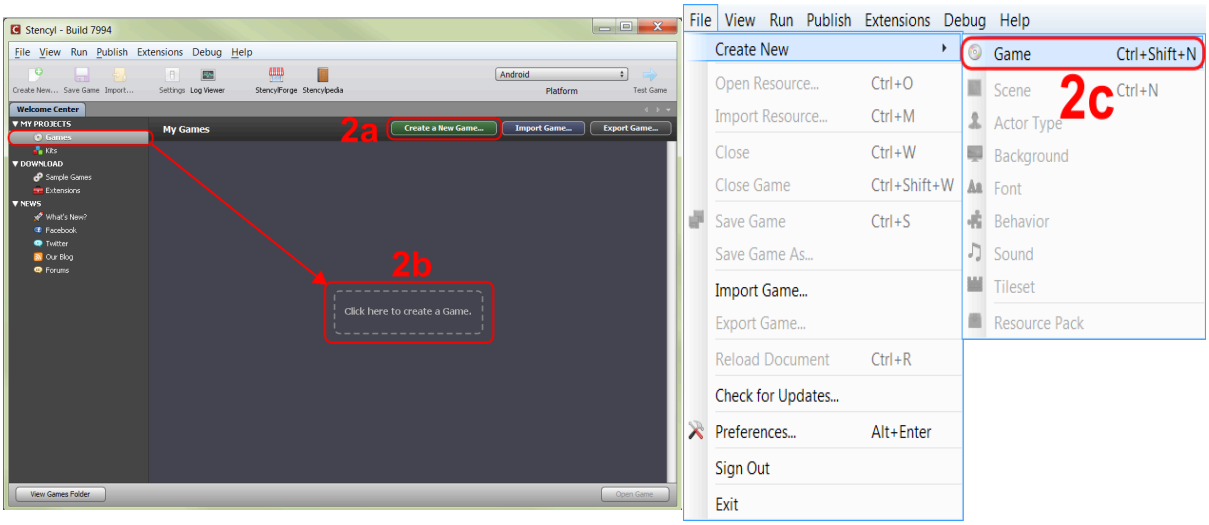
Instalace programu probíhá standardním způsobem (postupnými kroky se zobrazováním dialogových oken s umístěním instalované aplikace na lokálním disku) jako u většiny aplikací, proto komukoliv, kdo někdy nějaký program instaloval, nebude cizí ani postupinstalace tohoto programu.
Po prvním spuštění programu se objeví dialogové okno s možností vytvoření uživatelského účtu (obrázek **1a**), které po zadání identifikačních informací o uživateli (obrázek **1c**) umožňuje ukládat realizované projekty vytvořené v aplikaci prostřednictvím cloudového úložiště. Registrace může být odložena tlačítkem připomenout později (obrázek **1b**).



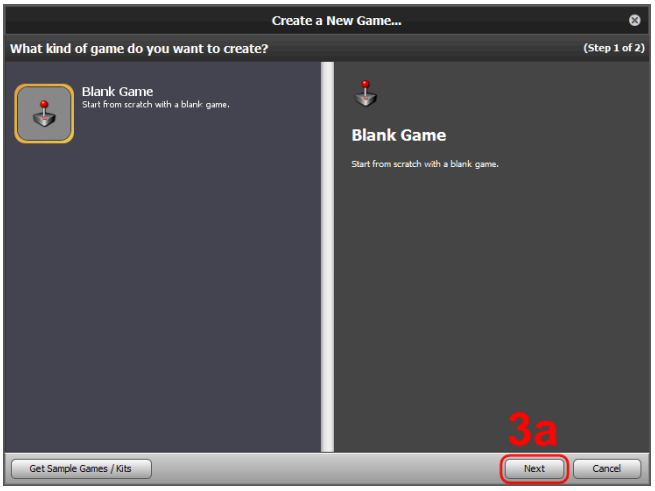
Registraci lze provést také přímo na webové stránce produktu Stencyl: www.stencyl.com/register

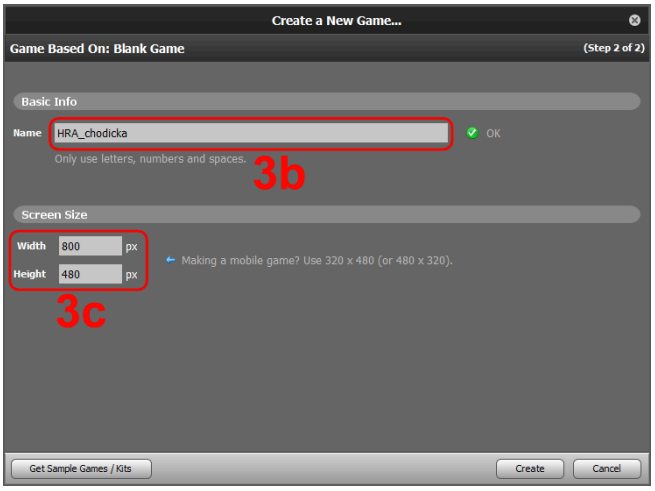
4. Vytvoření (založení) nové hry

Možností, jak vytvořit (založit) projekt s novou hrou, je hned několik. Zeleným tlačítkem (obrázek **2a**), kliknutím do oblasti ohraničené čárkovanou čarou uprostřed pracovní plochy (obrázek **2b**) - v takovém případě je nutné mít označenou položku **Games** v levé nabídce **My products**. Nebo je možné projekt s novou hrou vytvořit klasicky prostřednictvím hlavní lišty položkou **File** → **Create New** → **Game** (obrázek **2c**).



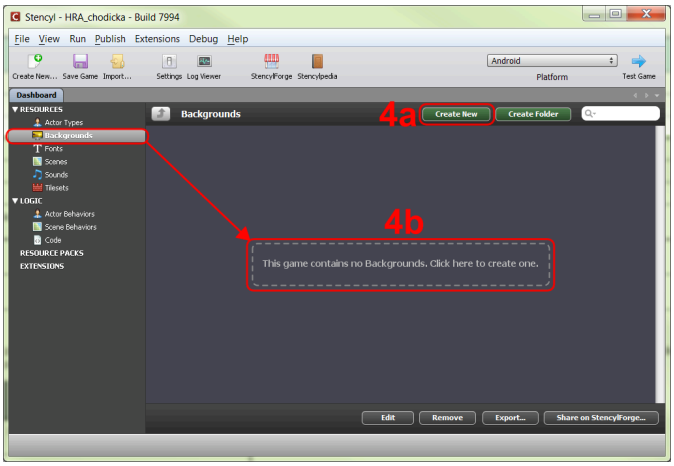
V úvodu vytvoření projektu s novou hrou je možné tlačítkem **Get Simple Games / Kits** vybrat z již hotových volně dostupnýchherních projektů, které byly realizovány komunitou Stencylvybrat nebo vytvořit prázdný projekt s hrou **Blank Game** od začátku a pokračovat (obrázek **3a**).
V dalším kroku po výběru prázdného projektu je vyžadováno vyplnění položek s názvem hry (obrázek **3b**) a nastavení rozměrů (šířky a výšky) herní obrazovky (obrázek **3c**).
Tlačítem **Create** se vytvoří projekt s dříve nastavenými základními parametry hry.

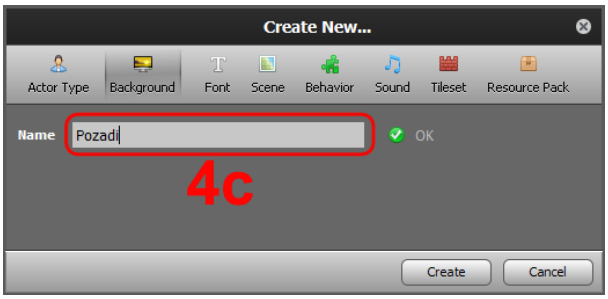




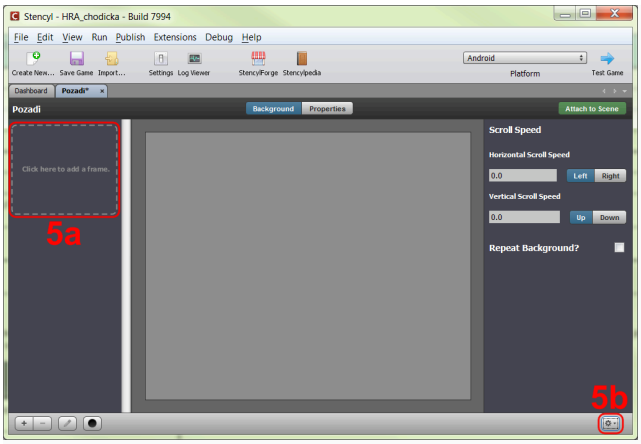
5. Příprava pozadí hry

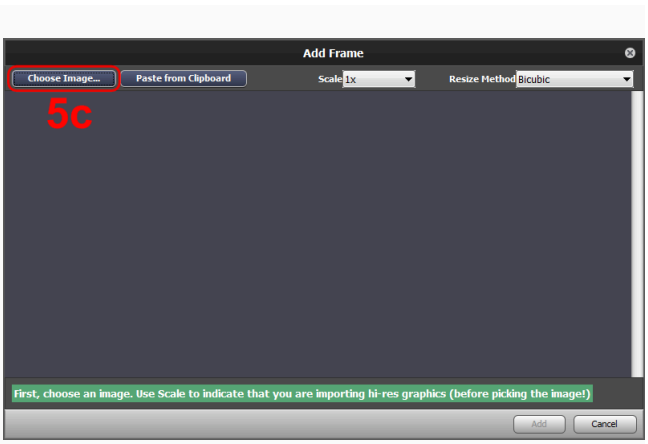
K vytvoření pozadí ve hře je nutné si připravit vhodný obrázek nebo barevnou plochu, na které dobře vyniknou (budou v kontrastu) ostatní herní prvky, jako hlavní hrdina, herní bloky (reprezentující zem, po které se bude hrdina pohybovat) a bonusy, které budehrdina sbírat.
Příprava položky reprezentující obrázek na pozadí může být provedeno tlačítkem (obrázek **4a**) nebo kliknutím do oblasti ohraničenéčárkovanou čarou uprostřed pracovní plochy (obrázek **4b**) - v takovém případě je nutné mít označenou položku **Backgrounds** v levé nabídce **Resources**.
Při pojmenování položky reprezentující pozadí (obrázek **4c**) stejně jako i další položky v rámci projektu musí být pojmenovány bez diakritiky.





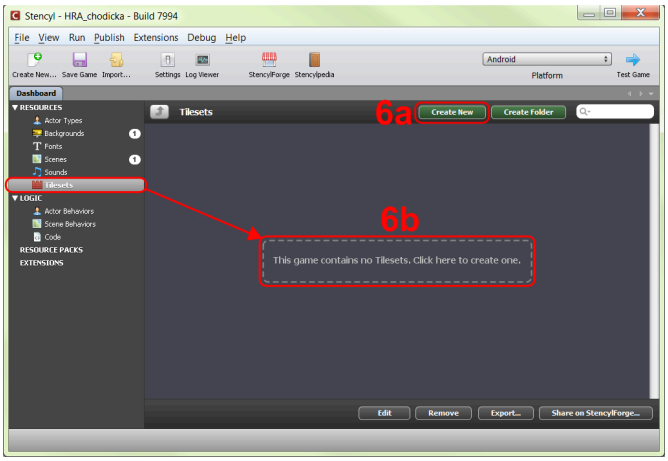
Snímek herního pozadí může být přidán kliknutím do oblasti ohraničené čárkovanou čarou v levé části pracovního okna (obrázek **5a**) nebo tlačítkem s ikonou ozubeného kola v pravém dolním rohu pracovního okna (obrázek **5b**) a následně volbou **Add Frame**. V dialogovém okně s názvem Add Frame pomocí tlačítka **Choose Image** (obrázek **5c**) se otevře okno s možností procházení souborůna lokálním disku a dohledat připravený obrázek pozadí už bude snadné.
Po výběru připraveného obrázku pozadí se rozsvítí tlačítko **Add** v pravém dolním okraji dialogového okna. Tím se obrázek přidá jako snímek, který pak bude viditelný v pracovním okně nejen na pozici **5a**, ale také bude evidován pod položkou levé postrannínabídky **Backgrounds** na záložce **Dashboard**.

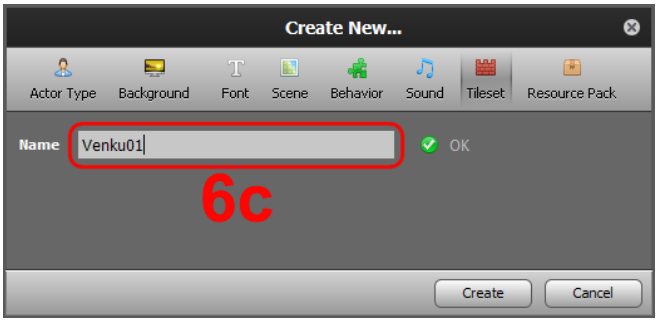




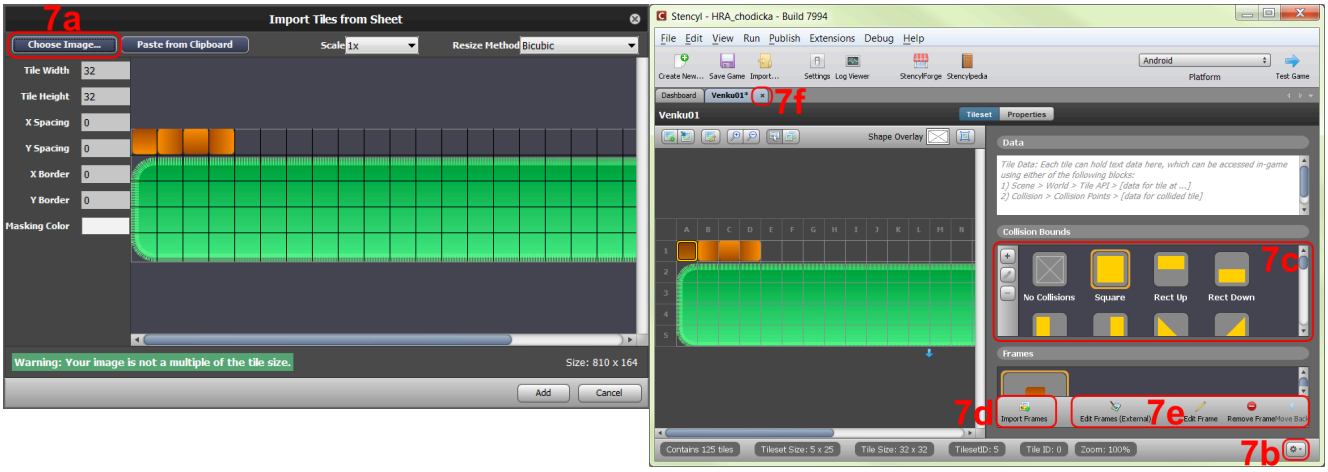
6. Realizace herního prostředí

Prostředím hry jsou myšleny především nejrůznější plochy, překážky a bariéry v popředí, po kterých se může herní postava pohybovat nebo kterým se má vyhýbat. Stejně jako v případě obrázku na pozadí, tak i objekty herního prostředí je nutné do projektu vložit. V případě přípravy zcela vlastní hry bude nutné vyvtvořit v nějakém grafickém editoru rastrové obrázky reprezentující herníprostředí. Pokud při prvotním návrhu obrázků na nějaké prvky herního prostředí člověk zapomene, vždy se dají přidávat další.
Do projektu hry se položky herního prostředí vkládají obdobně jako v případě herního pozadí a jiných herních prvků, tedy tlačítkem (obrázek **6a**) nebo kliknutím do oblasti ohraničené čárkovanou čarou uprostřed pracovní plochy (obrázek **6b**) - v obou případech jenutné mít označenou položku **Tilesets** v levé nabídce **Resources**.
Jako u jiných položek, tak i zde je nutné položku projektu pojmenovat bez diakritiky (obrázek **6c**).





Nyní je možné z lokálního disku vybrat připravený obrázek s herními prvky prostředí (obrázek **7a**) a pomocí tlačítka **Add** přidat vytvořenou položku.
Před návratem na hlavní stránku projektu se může s obrázkem herního prostředí dále pracovat a mohou být nastaveny jeho vlastnosti, jako např. tvar oblasti kolize s objektem (obrázek **7c**). Oblast kolize je důležitý herní aspekt, který je nastavován nejen u herního prostředí, ale také u herních postav. Určuje reakční oblast daného objektu, tedy v případě prostředí musí být určena oblast,po které se herní postava pohybuje (případně do které herní postava naráží).
Další možnosti práce s obrázky prostředí jsou např. přidávání dalších obrázků prostředí v rámci jedné položky tlačítkem **ImportFrames** (obrázek **7d**) či tlačítkem s ikonou ozubeného kola (obrázek **7b**) a pak položkou **Add Tiles**, dále je možné editovat (popř. odstraňovat) stávající obrázky prostředí jako celek nebo jen jednotlivé dílky (obrázek **7e**).
Uložení všech úprav se provede automaticky po zavření záložky s obrázky herního prostředí (obrázek **7f**).



7. Realizace herních postav

Na podobném principu vytváření položek připravených obrázků (uložených na lokálním disku) fungují také další herní prvky, jako herní postavy **Actor Types**.
Úkol: Zamyslete se nad tím, jak se budou jednotlivé herní postavy (hrdina, se kterým pohybuje hráč, a záporné postavy) pohybovat (směry pohybu, výskoky apod.), vytvořte obrázky těchto postav v různých polohách (pohled postav podle směru pohybu, pohyb, výskok apod.) a vložte svoje obrázky do herního prostředí k jednotlivým postavám. Jednotlivé typy postav by v projektu měly bývtvidět jako položky. Také si uvědomte, že případné bonusy, které bude "hrdina" sbírat, jsou také jedním typem herní postavy.

8. Realizace herní scény

Dříve navržené herní prvky (pozadí, prostředí a postavy) bude možné zkombinovat v prostředí herní scény.
Úkol: Vytvořte položku herní scény s názvem Level01 a navrhněte prostředí 1. kola hry tím, že vložíte dříve připravené herní prvky do scény (prvky najdete v sekcích návrhu scény nazvaných jako **Tiles** a **Actors**). Pozadí hry vložte jako novou vrstvu umístěnou zcela dole. Připravenou scénu otestujte na platformě Flash (Player) tlačítkem Test Scene.

6 Závěrečné poznámky a zdroje

Zdroje multimediálního materiálu

Obrázky:

Kapitola 3:

Obrázek stolního PC pod CC0 public domain licencí použit ze zdroje:

http://pixabay.com/en/computer-desktop-modern-device-154114/

Obrázek notebooku pod CC0 public domain licencí použit ze zdroje:

http://pixabay.com/en/laptop-notebook-mobile-computer-154091/

Obrázek tabletu byl vytvořen autorem kurzu.

Loga jednotlivých aplikací a vývojových prostředí byla získána z oficiálních stránek jednotlivých popisovaných produktů.

Kapitola 5:

Obrázky demonstrující postup práce byly autorem kurzu nasnímány a doplněny o popisky ve volně dostupných grafických editorech.

Videa:

Kapitola 2:

Motivační videa demonstrující přípravu principiálně podobné Android aplikace ve třech různých prostředích byla použita z video-serveru youtube:

Flash projekt:

https://www.youtube.com/watch?feature=player_embedded&v=_jZBe1KGhnc Stencyl

projekt: https://www.youtube.com/watch?feature=player_embedded&v=CCt-8i4Cnqk

GameSalad projekt:

https://www.youtube.com/watch?feature=player_embedded&v=xIlabLJf8To

Kapitola 4:

Videa demonstrující postup práce a ukázky finálních produktů byla zaznamenána autorem pomocí programu Adobe Captivate dostupného na katedře výpočetní a didaktické techniky na FPE ZČU v Plzni pouze pro účely tohoto kurzu.

Programovací prostředí EV3 a RobotC robotické stavebnice LEGO Mindstorms EV3

Kapitola se věnuje programovacím prostředím EV3 a RobotC, která jsou určená k programování robotické stavebnice EV3. Popisuje samotná prostředí, vysvětluje princip vytváření programu v těchto dvou prostředích a popisuje také programovací jazyk k tomu určený. Seznámíte se s důležitými funkcemi EV3 a RobotC, které napomáhají k efektivnější práci s programem nebo jsou nezbytné pro vytváření programu či práci s řídící jednotkou stavebnice. Kapitola si klade za cíl usnadnit Vám prvotní využívání některého z těchto dvou prostředí ve výuce.

Využité přístroje a software:

programovací prostředí EV3 a RobotC

robotická stavebnice LEGO Mindstorms EV3

Cílová skupina/náročnost: 1. až 4. ročník SŠ a odpovídající ročníky gymnázií

Všechny uvedené texty, obrázky a videa jsou vlastní, není-li uvedeno jinak. Autory Youtube embed videí lze nalézt při kliknutí na znak Youtube ve videu během přehrávání.

Autor:

Mgr. Jan Baťko

K plnohodnotnému využití této studijní opory je nutný přístup k on-line zdrojům a materiálům.

Tento materiál vznikl z finanční podpory Evropského sociálního fondu a státního rozpočtu České republiky v rámci projektu „Popularizace vědy a badatelsky orientované výuky“, reg .č. CZ.1.07/2.3.00/45.0007.

Programovací prostředí EV3

1 Základní informace o projektu

Název

Programovací prostředí EV3

Anotace programu/zaměření/hlavní cíl

Programovací prostředí EV3 je ikonické programovací prostředí určené pro vytváření programů pro robotickou stavebnici LEGO Mindstorms EV3. Hlavním cílem této kapitoly je seznámit čtenáře nejprve se samotným programovacím prostředím a následně demonstrovat jeho funkce a možnosti pomocí jednoduchých aktivit přiřazených do některých kapitol.

Cílová skupina

1. až 4. ročník SŠ a odpovídající ročníky gymnázií

Organizační podmínky

Spolupráce studentů ve dvoučlenných, maximálně tříčlenných skupinách. Pro výuku je vhodné využít běžnou učebnu vybavenou počítači s nainstalovaným programovacím prostředím EV3.

Pomůcky

Robotická stavebnice LEGO Mindstorms EV3, programovací prostředí EV3.

Časová náročnost

Odhadovaná doba: zhruba 15 - 20 vyučovacích hodin.

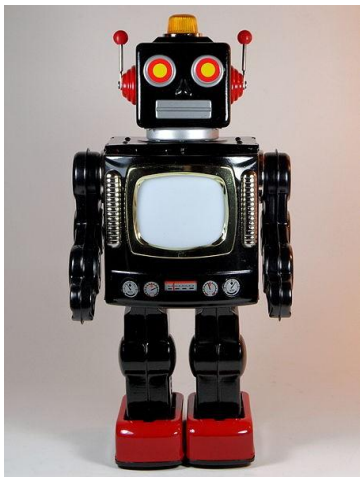
Vazba na RVP

Rámcový vzdělávací program pro gymnázia.

Mezipředmětové vazby

Informatika (informační a komunikační technologie), elektrotechnika

2 Motivační rámec projektu



Text:

Všichni jistě znáte zkratku RUR (Rossumovi univerzální roboti). Zkratku, kterou ve svém stejnojmenném díle použil spisovatel Karel Čapek. V knize varoval před možnými negativními vlivy techniky na život člověka. My již dnes ale víme, že technika nám práci v mnohých případech hlavně usnadňuje. Navíc je nám také známo, že pokud se robot chová tak, že by ohrozil například naše zdraví, je to většinou chybou člověka. Špatnou manipulací, nedůslednou tvorbou nebo chybou ovládacího programu.

Při svém bádání s robotickou stavebnicí LEGO Mindstorms EV3 se ale žádného nebezpečí bát nemusíme. Veškeré chování robota máme pevně v rukou. V tomto modulu se dozvíte všechny podstatné informace týkající se robotického programovacího prostředí EV3. Jedná se o ikonické programovací prostředí umožňující vytváření robotických ovládacích programů. Při jeho používání nemusíte znát syntaxi žádného programovacího jazyka. Program se v něm vytváří pouze za pomoci logického uspořádávání a propojování programových bloků. Veškeré získané znalosti si můžete upevnit a ověřit splněním jednotlivých aktivit, které jsou obsaženy v každé kapitole.

Zdroj obrázku: commons.wikimedia.org, autor: D J Shin, BY-SA

Doporučený multimediální materiál

EV3 (home verze) ke stažení na oficiální stránkách LEGO: stahujte ZDE (odkaz viz. on-line kurz)

Ilustrační videa k demonstraci možností EV3: více ZDE (odkaz viz. on-line kurz)

Přehled možností EV3: více ZDE (odkaz viz. on-line kurz)

3 Poznámky k využití přístrojů

Pro tvorbu úvodu do práce s programovacím prostředím EV3 a sestavení úloh byla použita základní sada stavebnice EV3. Další informace o ní nalezenete na následujících odkazech:

Informace na oficiálních internetových stránkách výrobce: ZDE (odkaz viz. on-line kurz)

Nabídka základních i doplňkových komponent na prodejním webu české společnosti EDUXE s.r.o.: ZDE (odkaz viz. on-line kurz)

Výrobce uvádí na svých stránkách také několik informací k programovacímu prostředí EV3. Naleznete je ZDE. (odkaz viz. on-line kurz)

4 Projektový deník

Projektový deník slouží žákům k evidenci svého postupu v kurzu programování v EV3. K ověření znalosti každé části slouží příslušná aktivita, kterou by měl žák vyřešit. Při úspěšném vyřešení je proveden zápis do projektového deníku. Záznam by měl obsahovat poznámku o tom, kdy byla úloha zpracovávána, a poté krátký popis postupu a problémů, které bylo při tvorbě potřeba řešit. Po vyřešení úlohy vyučující zkontroluje funkčnost a správnost konstrukce nebo vytvořeného programu a zapíše do deníku hodnocení.

Projektový deník ke stažení v kurzu, zároveň jej najdete jako přílohu této tiskové opory.

5 Základní orientace v programovacím prostředí EV3

Programovací prostředí EV3 si můžeme rozdělit do několika částí, které si nyní popíšeme. Pro ilustraci je můžete najít vyznačené v následující animaci. (video viz. on-line kurz)

Popis jednotlivých částí EV3

Horní menu programu:

- obsahuje základní volby pro práci se souborem (vytvoření nového programu nebo projektu, uložení, editace),
- zahrnuje rozšiřující možnosti programovacího prostředí (editor zvuku a obrázků, tvůrce vlastních bloků, aktualizaci firmware, nastavení bezdrátového připojení, import nových programových bloků nebo správce paměti).

Programovací plocha:

- zabírá největší část programovacího prostředí,
- slouží k logickému uspořádávání a propojování programových bloků, začátek programu je zde vyznačen blokem Start.

Navigační tlačítka a orientace v programu:

- tlačítka umístěná vpravo nad programovací plochou,
- nalezneme zde výběrová tlačítka, vkládání komentářů, rychlou volbu pro ukládání, tlačítka pro krok zpět či vpřed nebo možnosti přiblížení a oddálení programovací plochy.

Knihovna programových bloků:

- umístěna pod programovací plochou, obsahuje všechny programovací bloky,
- rozdělena do šesti barevně odlišených kategorií (poslední obsahuje vlastní vytvořené bloky).

Správa připojených senzorů a modulů, informace o řídící jednotce:

- okno umístěné v programovacím prostředí z pohledu uživatele vpravo dole,
- obsahuje informace o řídicí jednotce (verze firmware, zaplnění paměti), zařízeních připojených k řídicí jednotce, dostupné řídicí jednotky (připojené pomocí USB, Wi-fi nebo Bluetooth).

Tlačítka pro nahrání programu do řídicí jednotky:

- umístěná vpravo od okna pro správu řídicí jednotky,
- jedná se o volby buďto pouze pro stažení programu do řídicí jednotky, nebo i jeho okamžité spuštění,
- třetí tlačítko slouží k testování zvolené části programového kódu (po označení spustí zvolený úsek programového kódu).

5.1 Aktivita 1 - Příprava projektu

Téma	Příprava projektu v EV3	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Kdykoli vytváříte nějaký program, je dobré, abyste si jej vhodně pojmenovali, popsali, případně doplnili zdrojový kód o komentáře. Programovací prostředí EV3 umožňuje celý projekt doplnit o fotografie, videa či vlastní popis. Díky tomu i po delším čase, po kterém projekt otevřete, na první pohled zjistíte, jaká je funkce jednotlivých programů. V této aktivitě si přípravu nového projektu a jeho popis vyzkoušíte.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3, digitální fotoaparát nebo mobilní telefon.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve z robotické stavebnice sestaví vlastní model pojízdného robota pro úlohu zabývající se možnostmi pohybu pojídzného robota pomocí servomotorů. Pro tuto úlohu si vytvoří v programovacím prostředí EV3 nový projekt a příslušné programy, které si vhodně pojmenují. Následně pomocí digitálního fotoaparátu nebo mobilního telefonu pořídí fotografie svého modelu. Ty následně přidají do projektu v EV3, který si doplní vhodným popisem programu a dalšími komentáři.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci budou schopni vytvořit nový projekt v programovacím prostředí EV3, doplnit ho o popisné informace, komentáře a multimediální prvky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Nejsou potřeba žádné vstupní znalosti.	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Pořízení a úprava fotografií.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Popis a tvorba projektu v programovacím prostředí EV3.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

V této aktivitě se naučíte vytvořit a vhodným způsobem si připravit svůj první projekt v EV3. Příprava projektu je prvním krokem, který musíte absolvovat při vytváření nového programu. Zároveň jsou to také první úkony, díky kterým se zevrubně s programovacím prostředím seznámíte. Pro úspěšné absolvování tohoto úkolu musíte splnit toto zadání:

1. Sestavte z robotické stavebnice EV3 model pojízdeného robota, který je poháněn dvěma servomotory.
2. Následně vytvořený model vhodně nafoťte pomocí digitálního fotoaparátu nebo mobilního telefonu. Nezapomeňte na to, že pořízené fotografie by měly vhodně představovat konstrukci robota případnému uživateli, který bude s programem v budoucnosti pracovat.
3. Založte si v programovacím prostředí nový projekt, uložte jej, vhodně pojmenujte jak projekt, tak dílčí program. Ve vlastnostech projektu stručně popište, k čemu projekt slouží, a doplňte jej o fotografie sestaveného modelu.

6 Základní výstupní moduly

Kapitola se věnuje základním výstupním modulům robotické stavebnice LEGO Mindstorms EV3. Výstupní zařízení slouží k různým možnostem signalizace či přenášení různé hnací síly navenek. V kapitole se věnujeme následujícím modulům:

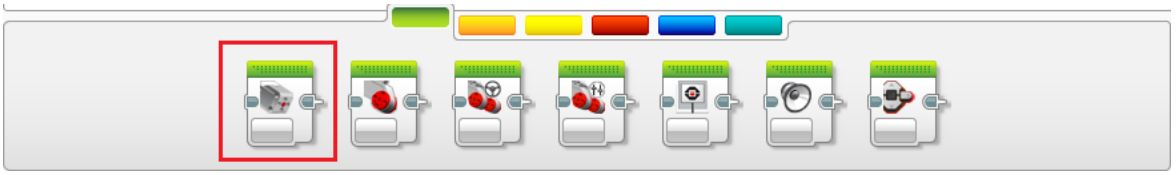
- Střední motor
- Velký servomotor
- Možnosti řízení vice motorů
- Světelná signalizace řídící jednotky
- Výstup na displej

6.1 Střední motor

Možnosti použití středního motoru

Umístění

Blok Medium Motor pro ovládání středního motoru je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy chodu středního motoru

Nastavení výstupního portu řídicí jednotky (A - D), ke kterému je motor připojen, provedeme v malém okně v pravém horním rohu programového bloku.

Vypnuto (Off)

Režim, který slouží k zablokování chodu motoru. Programový blok v tomto režimu obsahuje pouze jediný vstupní port, u kterého volíme, zda se má motor při pokynu k zastavení zastavit okamžitě (Break at End: True), nebo s pozvolným mírným dojezdem (Break at End: False).

Zapnuto (On)

Při použití programového bloku v tomto režimu dáváme motoru pokyn k neustálému otáčení zvolenou rychlostí (0 - 100 %), kterou nastavíme u jediného vstupního portu.

Otáčení po určitou dobu (On for Seconds)

Režim, který umožňuje nastavit otáčení motoru po určitou dobu stanovenou v sekundách. U prvního vstupního portu nastavíme požadovanou rychlost otáčení, u druhého zmíněný časový úsek a u třetího, zda se má motor na konci otáčení okamžitě zastavit (Break), nebo plynule dojet (Coast).

Otočení o určitý úhel (On for Degrees)

Režim, u kterého se míra natočení motoru stanovuje pomocí úhlových stupňů (0 - 360°). Zbývající dva vstupní porty jsou totožné jako u režimu pro otáčení po určitou dobu. Jedná se o nastavení rychlosti otáčení a způsob zastavení motoru (Break nebo Coast).

Otočení o určitý počet otáček (On for Rotations)

Režim je totožný se dvěma předchozími. Ovšem míra natočení se zde udává pomocí počtu nastavených otáček kolem své osy.

6.2 Aktivita 2 - Lopatková turbína

Téma	Lopatková turbína	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Trendem dnešní doby je využívání obnovitelných zdrojů energie. Zásoba přírodních zdrojů není nevyčerpatelná, a tak musíme stále hledat vhodné alternativy. Během slunečných dnů můžeme využívat například solární panely. Jak ovšem vyrábět elektřinu ve stejném objemu i v období, kdy není světla dostatek nebo v noci? V této úloze si sestrojíte jednoduchou turbínu fungující jako záložní způsob výroby energie v situacích, kdy je světla nedostatek.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve vytvoří jednoduchý model lopatkové turbíny. Ta bude poháněna pomocí střední motoru. Její chod bude řízen barevným senzorem, který pomocí detekce okolního světla bude řídit rychlost otáčení.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat střední motor k pohonu rotačních součástí a řídit rychlost otáčení v závislosti na jiném zařízení.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s barevným senzorem.	
Mezipředmětové vztahy	Fyzika (světlo), informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Vytvořte funkční model turbíny jako záložního zdroje výroby elektrické energie. Požadavky na konstrukci modelu a program budou následující:

1. Lopatky turbíny budou poháněny středním motorem.
2. Řízení rychlosti otáčení bude prováděno díky barevnému senzoru.
    - a. Čím nižší bude intenzita okolního světla, tím se bude turbína otáčet rychleji.
3. V programu vhodně nastavte prahovou hodnotu dostatečného slunečního svitu, při kterém se turbína zastaví.

Výsledný program ke stažení ve formátu .ev3

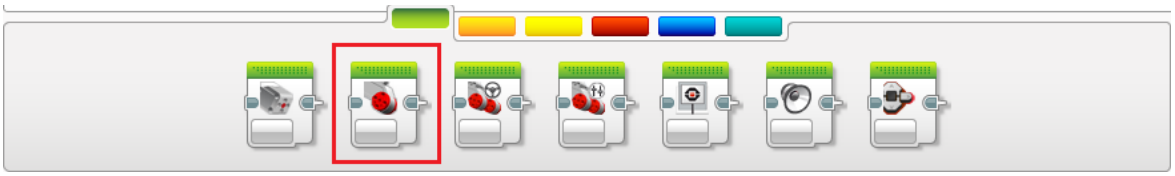
Program ke stažení v on-line kurzu

6.3 Velký servomotor

Možnosti použití velkého motoru

Umístění

Blok Large Motor pro ovládání velkého motoru je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy chodu vekého motoru

Nastavení výstupního portu řídicí jednotky (A - D), ke kterému je motor připojen, vybereme v malém okně v pravém

horním rohu programového bloku.

Vypnuto (Off)

Režim, který slouží k zablokování chodu motoru. Programový blok v tomto režimu obsahuje pouze jediný vstupní port, u kterého volíme, zda se má motor při pokynu k zastavení zastavit okamžitě (Break at End: True), nebo s pozvolným mírným dojezdem (Break at End: False).

Zapnuto (On)

Při použití programového bloku v tomto režimu dáváme motoru pokyn k neustálému otáčení zvolenou rychlostí (0 - 100 %), kterou nastavíme u jediného vstupního portu.

Otáčení po určitou dobu (On for Seconds)

Režim otáčení motoru po určitou dobu zajišťuje otáčení po dobu, která se nastavuje na druhém vstupním portu v sekundách. Další volbou je u tohoto režimu rychlost otáčení udávaná v procentech (0 - 100 %) a způsob zastavení motoru. Pro okamžité zastavení volíme možnost Break, pro pozvolný dojezd motoru možnost Coast.

Otočení o určitý úhel (On for Degrees)

Režim On for Degrees funguje naprosto stejně jako předchozí režim On for Seconds. Místo doby otáčení nastavené v sekundách zde ovšem volíme míru natočení zadávanou ve stupních (0 - 360°). Všechny ostatní volby jsou totožné.

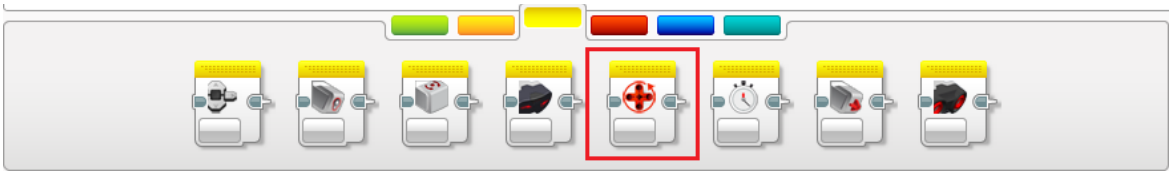
Otočení o určitý počet otáček (On for Rotations)

Režim natočení o určitý počet otáček je opět totožný jako předchozí. Míra natočení motoru se v tomto režimu udává jako počet provedených otáček kolem osy motoru.

Možnosti získávání dat z motorů

Umístění

Blok Motor Rotation pro získávání dat ze servomotoru je umístěn ve žluté záložce Sensor (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy měření dat

Nastavení výstupního portu řídicí jednotky (A - D), ke kterému je motor připojen, vybereme v malém okně v pravém horním rohu programového bloku.

Měření stupňů natočení (Measure Degrees)

Režim, ve kterém výstupní port bloku vrací změřené natočení motoru v úhlových stupních.

Měření otáček (Measure Rotations)

Režim, ve kterém výstupní port bloku vrací změřený počet otáček motoru.

Měření rychlosti otáčení (Measure Current Power)

Režim, ve kterém výstupní port bloku vrací rychlost otáčení motoru od -100 do 100.

Porovnávání stupňů natočení (Compare Degrees)

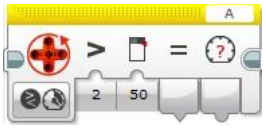
Blok v tomto režimu nejen vrací počet stupňů natočení motoru, ale také umožňuje porovnávat zjištěnou hodnotu s hodnotou pevně zadanou (Treshold Value). Na výstupním portu Result navíc vrací hodnotu logického datového typu, který vyjadřuje, zda byla splněna podmínka nastavená pro porovnání (True), nebo nikoliv (False).

Porovnávání počtu otáček (Compare Rotations)

Blok v tomto režimu pracuje totožně jako v režimu pro porovnávání natočení v úhlových stupních. V tomto režimu ovšem pracujeme s počtem otáček.

Porovnávání rychlosti otáčení (Compare Current Power)

Režim bloku, který je totožný s oběma předchozími. Pouze pracuje s rychlostí otáčení motoru.



Reset (Nulování)

Režim, který nastavuje čítač všech tří měřených veličin (natočení ve stupních, otáčkách, rychlost) na nulu.

6.4 Aktivita 3 - Kuchyňská minutka

Téma	Kuchyňská minutka	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Pravděpodobně každý z nás se někdy v kuchyni setkal s kuchyňskou minutkou, malým zařízením sloužícím k měření času při přípravě pokrmů. Využívaly ji naše babičky i maminky. Doba pokročila a z klasických manuálních se vyvinuly i pokročilejší digitální, které jsou často opatřené displejem. Všechno potřebné k tomu, abychom podobnou minutku vytvořili z robotické stavebnice, ovšem máme k dispozici i my. Postačí nám k tomu displej řídicí jednotky a otočný mechanismus zajistí servomotor.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve vytvoří vhodnou konstrukci kuchyňské minutky sestávající ze servomotoru opatřeného dobře ovladatelným otáčecím mechanismem pro zajištění pohodlného nastavení času měření. Odpočítávání začne ve chvíli, kdy bude stisknuto tlačítko připevněného dotykového senzoru.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat servomotor k ovládání pohyblivých součástí a pracovat s měřením a ovlivňováním vykonaného počtu otáček.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s dotykovým senzorem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
30 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
10 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujících multimediálních prvků.	

Zadání

Vytvořte funkční model kuchyňské minutky. Požadavky na konstrukci modelu a program budou následující:

1. Otočný mechanismus minutky bude realizován pomocí servomotoru.
2. Natočením mechanismu se díky detekci počtu otáček vyhodnotí, jak dlouhý časový úsek se má měřit.
3. Nastavovovaný čas by se měl zobrazit na displeji.
4. Při měření času by se měla minutka otáčet a konec odpočítávání by měl oznámit zvukový signál.

Výsledný program ke stažení ve formátu .ev3

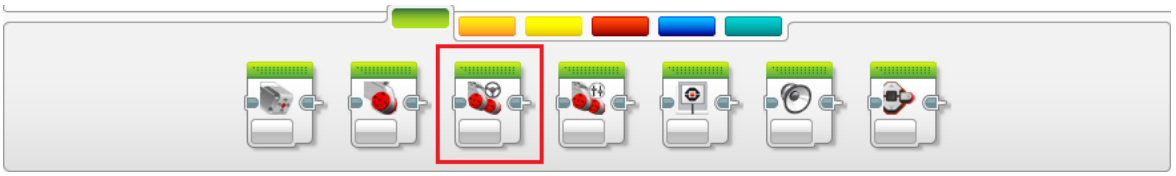
Program ke stažení v on-line kurzu

6.5 Řízení více motorů

Možnosti řízení robota pomocí dvou motorů

Umístění

Blok Move Steering pro řízení dvou servomotorů je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy bloku Move Steering

Nastavení výstupních portů řídicí jednotky (A - D), ke kterým jsou motory připojeny, provedeme v malém okně v pravém horním rohu programového bloku. Na rozdíl od řízení jediného motoru blokem Large Motor zde volíme dva výstupní porty.

Vypnuto (Off)

Jedná se o režim řízení, který zastaví motory, které byly do té doby v chodu. U vypnutí můžeme zvolit dva způsoby zastavení. Break, který motory okamžitě zastaví, a Coast, který umožní volné a setrvačné dojetí motorů.

Zapnuto (On)

Režim, který umožňuje souběžné otáčení motorů vpřed. Pomocí vstupního portu Steering je možné řídit směr, kterým se bude robot pohybovat (jiný než přímý). Druhý vstupní port (Power) slouží k nastavení rychlosti otáčení na stupnici od 0 do 100 %.

Otáčení po určitou dobu (On for Seconds)

Režim, který je ve své podstatě totožný jako předchozí. Umožňuje ovšem řídit dobu otáčení motorů v sekundách pomocí vstupního portu Seconds. Druhou odlišností je možnost nastavení způsobu zastavení motorů (Break pro okamžité zastavení nebo Coast pro volný dojezd).

Otočení o určitý úhel (On for Degrees)

Režim s totožnými možnosti jako režim On for Seconds. Místo nastavení doby pohybu ovšem využívá míru natočení motorů zadávanou ve stupních od 0 do 360°.

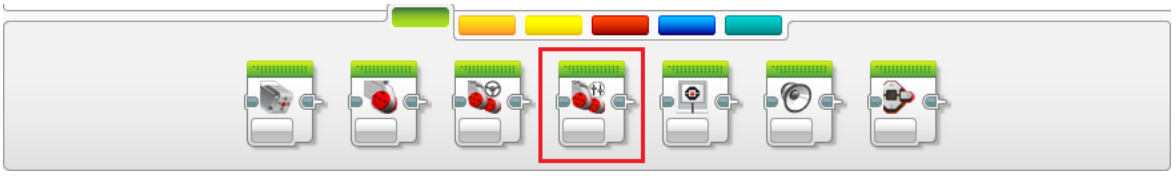
Otočení o určitý počet otáček (On for Rotations)

Třetí režim umožňující ovlivňovat dobu pohybu motorů. Oproti předchozím možnostem je zde nastavována pomocí počtu otáček motoru. Ostatní volby jsou totožné jako u předchozích režimů.

Možnosti řízení robota na bázi pásového vozidla

Umístění

Blok Move Tank pro řízení pásového vozidla je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Vypnuto (Off)

Režimy bloku pro řízení pásového vozidla se v základu shodují s blokem pro řízení klasického robota řízeného dvěma motory (Move Steering). Režim prozastavení (Off) zastaví oba připojené motory. Pomocí jediného vstupního portu můžeme volit způsob zastavení (Break pro okamžité zastavení nebo Coast pro volné dojetí).

Zapnuto (On)

Režim pro zapnutí a nekonečné otáčení motorů je již uzpůsoben řízení pásového vozidla. Obsahuje dva vstupní porty umožňující nastavit rychlost otáčení každému motoru zvlášť. Hodnota se pohybuje od -100 % do 100 %. Záporná hodnota vyjadřuje otáčení vzad, kladná poté otáčení vpřed.

Otáčení po určitou dobu (On for Seconds)

Rozšířením přechozího režimu On je režim On for Second. Ten umožňuje řídit dobu otáčení motoru v sekundách. Navíc umožňuje také zvolit způsob zastavení robota (Break nebo Coast).

Otočení o určitý úhel (On for Degrees)

Režim, který umožňuje řídit otáčení motorů nastavením míry jejich otočení v úhlových stupních (0 - 360°).

Otočení o určitý počet otáček (On for Rotations)

Režim On for Rotations slouží k řízení pohybu pásového vozidla nastavením počtu otáček, které mají motory vykonat. Ostatní nastavení je totožné jako u ostatních režimů pro ovlivnění doby otáčení motorů.

6.6 Aktivita 4 - Řízení pásového transportéru

Téma	Rízení pásového transportéru	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Při vytváření pojízdných modelů máme dvě možnosti, jak robota řídit. Prvním z typů je klasické kolové vozidlo a druhý model je ovládaný pomocí pásů. Pro každý z modelů využívá programovací prostředí EV3 jiný blok. Pokuste se tedy vymyslet, jak by mohl být ovládán pásový transportér nebo tank oproti klasickému kolovému vozidlu podobnému osobnímu automobilu.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Žáci si neprve vytvoří model pojízdného robota opatřeného pásy. Robot se bude pomalu pohybovat vpřed a jeho jízda bude ovlivňována dvěma dotykovými senzory. Při stisknutí každého z nich se robot natočí na požadovanou stranu. Tlačítka tedy budou sloužit jako směrovače pro otáčení do stran.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat alternativní způsob pohonu servomotorů (pomocí pásů).	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost možností řízení motorů, práce s dotykovým senzorem a znalost práce s cykly a podmínkami.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
20 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocen bude sestavený model robota a úplnost a funkčnost vytvářeného programu.	

Zadání

Vytvořte funkční model robota poháněného pásy. Požadavky na konstrukci modelu a program budou následující: robot musí být

dobře manévrovatelný a pásy na modelu dobře upevněny,
natáčení robota do stran bude prováděno dvěma dotykovými senzory plnícími funkci joysticku, volte vhodnou rychlost otáčení motorů pro dobrou pohyblivost modelu.

Výsledný program ke stažení ve formátu .ev3

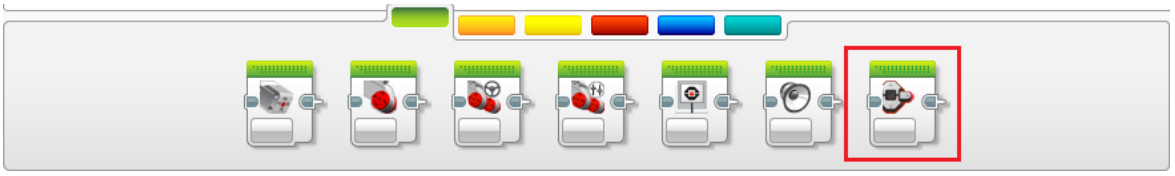
Program ke stažení v on-line kurzu.

6.7 Světelná a zvuková signalizace řídící jednotky

Možnosti použití světelné signalizace řídicí jednotky

Umístění

Blok Brick Status Light pro řízení podsvícení tlačítek řídicí jednotky je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



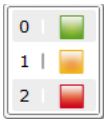
Režimy bloku pro řízení světelné signalizace řídicí jednotky

Vypnuto (Off)

Režim, který vypíná do té doby aktivní podsvícení řídicí jednotky.

Zapnuto (On)

Tímto režimem je možné zapnout světelnou signalizaci. Pomocí vstupního portu Color lze volit ze tří barevných provedení světelné signalizace, které můžete vidět na obrázku.



Vstupní port Pulse umožňuje spustit pulzování barevného podsvícení (True - spuštěno, False - vypnuto).

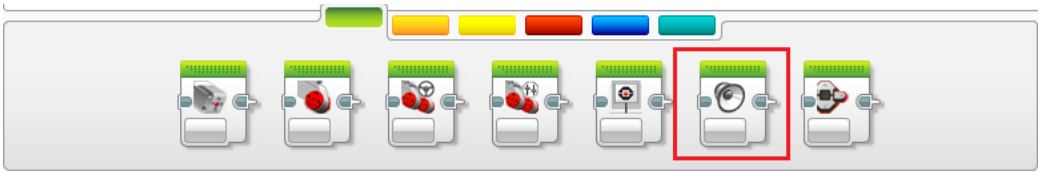
Základní nastavení (Reset)

Režim Reset nastaví barevné podsvícení zpět do základního nastavení, což je zelené blikající podsvícení indikující spuštěný program.

Možnosti použití zvukové signalizace řídicí jednotky

Umístění

Blok Sound pro řízení podsvícení tlačítek řídicí jednotky je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Zastavení přehrávání (Stop)

V tomto režimu blok zastaví přehrávání aktuálně přehrávaného zvuku prostřednictím řídicí jednotky EV3.

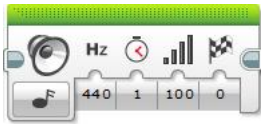
Přehrátí zvukového souboru (Play File)

Režim umožňuje přehrát zvukový soubor uložený v řídicí jednotce EV3. Vstupní port Volume (hlasitost) slouží k nastavení hlasitosti přehrávaného zvuku od 0 do 100 % a vstupní port Play Type (způsob přehrávání) vyjadřuje, jak má být soubor přehrán. K dispozici máme následující možnosti:

0 - Čekání na dokončení (Wait for Completion) - čeká na dokončení přehrávání zvuku a poté pokračuje dále. 1 - Přehrání jedenkrát (Play Once) - zvuk je jednou přehrán a poté program pokračuje okamžitě dále.
2 - Opakování (Repeat) - zvuk je přehráván tak dlouho, dokud není přerušen blokem v režimu Stop.

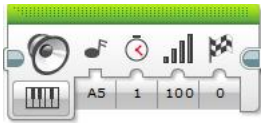
Přehrání tónu (Play Tone)

Režim pro přehrání tónu zvolené frekvence (Frequency) po určitou dobu trvání v sekundách (Duration) nastavenou hlasitostí. Možnosti přehrávání zvuku jsou totožné jako u předchozího režimu.



Přehrání noty (Play Note)

Režim pro přehrání zvolené hudební noty (Note) po určitou dobu v sekundách (Duration) zvolenou hlasitostí.Možnosti přehrávání zvuku jsou totožné jako u předchozího režimu.



6.8 Aktivita 5 - Železniční přejezd

Téma	Železniční přejezd	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Signalizace železničního přejezdu upozorňuje řidiče, že se k přejezdu blíží vlak, a je tudíž životu nebezpečné do něj vjíždět. Podoba železniční signalizace je různá. V této úloze si vytvoříte program, který bude simulovat funkci světelné signalizace na železničním přejezdu. Přidanou hodnotou bude zvuková signalizace upozrňující na blížící se nebezpečí.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je využít zvukovou a světelnou signalizaci řídicí jednotky jako signalizátor blížícího se nebezpečí na železničním přejezdu. Detektorem blížícího se vlaku se stane ultrazvukový senzor, který pomocí naměřené vzdálenosti spustí světelnou a zvukovou signalizaci.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat světelnou a zvukovou signalizaci řídicí jednotky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce se světelnou a zvukovou signalizací.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
40 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Využijte řídicí jednotku k vytvoření programu, který bude simulovat signalizaci blížícího se vlaku na železničním přejezdu. Požadavky na tvorbu budou následující:

1. Použijte ultrazvukový senzor pro detekci blížícího se vlaku.
2. Při naměření kritické vzdálenosti se spustí přerušovaná světelná a také zvuková signalizace.
3. Jakmile se pomyslný vlak vzdálí opět za kritickou vzdálenost, signalizace se vypne.

Výsledný program ke stažení ve formátu .ev3

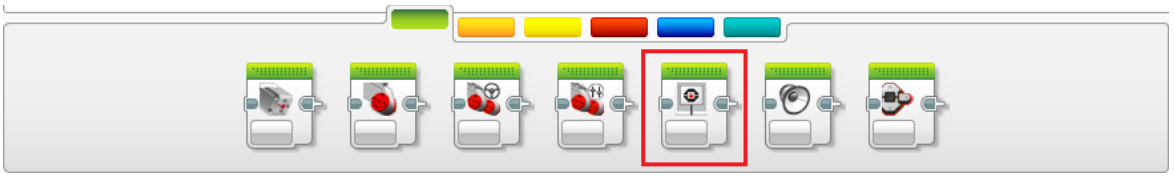
Program ke stažení v on-line kurzu.

6.9 Výstup na displej

Možnosti výstupu na displej řídicí jednotky

Umístění

Blok Display pro práci s displejem řídicí jednotky je umístěn v zelené záložce Action (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy programového bloku pro práci s displejem řídicí jednotky

Vypsání textu na pozici - v pixelech (Text - Pixels)

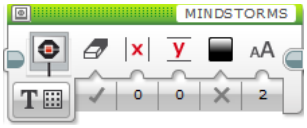
První ze dvou režimů umožňujících vypsání textu na displej umisťuje text na pozici konkrétního pixelu, který značí levý horní roh vypisovaného textového řetězce. Šířka displeje je 178 pixelů a výška 128 pixelů. Vstupní port X značí nastavení pozice v horizontální směru (-177 - +177). Vstupní port Y poté pozice ve vertikálním směru (-127 - +127). Volba Clear Screen umožňuje smazat displej před vypsáním textu. Volba Color umožňuje uživateli zvolit způsob vypsání textu. Pokud zvolíme Black, bude text vypsát černou barvou na bílé pozadí. Jestliže zvolíme White, bude vypsán bílým písmem na černé pozadí. U posledního vstupního portu Font můžeme nastavovat velikost vypisovaného textu (Normal, Bold nebo Large).

Požadovaný text k vypsání zapisujeme do pravého horního rohu bloku (na obrázku vzorově zapsán text MINDSTORMS). Tlačítkem Display Preview v pravém horním rohu bloku můžeme zobrazit náhled displeje a upravit tak případně pozici textu.

Po kliknutí na pole pro zápis textového řetězce je možné z menu vybrat volbu Wired. Díky tomu se na programovém bloku zobrazí vstupní port Text umožňující vypsat pomocí bloku Displej libovolný text přivedený na vstup pomocí vodiče.

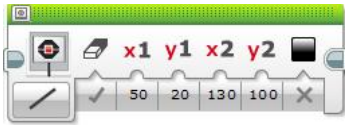
Vypsání textu na pozici - mřížka (Text - Grid)

Druhý režim pro výpis textu na displej obsahuje totožné volby jako režim Text - Pixels. Rozdíl je ovšem v pozici, na které je text umisťován. V tomto režimu není displej rozdělen na pixely, ale na mřížku tvořenou sloupci a řádky. Počátečním bodem výpisu je tedy buňka na zvolených souřadnicích. Displej je rozdělen na 22 sloupců po 8 pixelech (pozice X) a 12 řádků po 10 pixelech (pozice Y).



Vykreslení křivky (Shapes - Line)

Programový blok Display obsahuje čtyři režimy pro vykreslování obrazců. První z nich umožňuje vykreslovat křivky. Každá křivka je definována počátečním a koncovým bodem vykreslení. Souřadnice (x1, y1) určují počáteční pixel pro vykreslování a souřadnice (x2, y2) koncový pixel vykreslení. Stejně jako u vypsání textu umožňuje režim počáteční smazání displeje a volbu výpisu (černý text na bílém pozadí nebo bílý text na černém pozadí).



Vykreslení kruhu (Shapes - Circle)

Druhou možností pro vykreslení obrazců na displeji je zobrazení kruhu. Kruh se vykreslí na pozici, která je opět určena pixelem na souřadnicích X a Y. Pixel označuje střed kruhu. Druhým parametrem, který musíme nastavit, je poloměr kruhu (Radius). Ostatní možnosti jsou opět shodně s předchozími režimy.

Vykreslení čtverce nebo obdélníku (Shapes - Rectangle)

Režim pro vykreslení čtverce nebo na displej stejně jako předchozí pracuje s počátečními souřadnicemi pro vykreslení (x, y). Definují levý horní roh vykreslovaného obrazce. Následně musíme nastavit šířku (Width) a výšku (Height) obrazce udávanou v pixelech. Poslední charakteristickou volbou pro tento režim je možnost vykreslit obrazec s výplní (Fill).

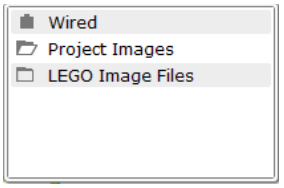
Vykreslení bodu (Shapes - Point)

Posledním režimem pro vykreslení obrazců na displej je režim umožňující vykreslit jediný bod (pixel) displeje řídicí jednotky. Pixel je opět definován souřadnicemi (x, y). Následně již pouze záleží, zda chceme vykreslovat bod(y) černě na bílém pozadí, nebo bíle na černém pozadí.

Vykreslení obrázku ze souboru (Image)

Na displej řídicí jednotky není možné vypisovat pouze text nebo obrazce pomocí propojených bodů (pixelů) displeje. Další možností je zobrazení obrázku uloženého v paměti řídicí jednotky. Obrázek vybereme po kliknutí na bílé pole v pravém horním rohu bloku.

Zobrazí se nám následující výběrové okno:



Zvolený obrázek následně vykreslíme na souřadnice (x, y). Pokud vybereme volbu Wired, zobrazí se na programovém bloku vstupní port File Name.

Smazání plochy displeje (Reset)

Režim, který umožňuje kompletně smazat plochu displeje řídicí jednotky.

6.10 Aktivita 6 - Kreslicí tabulka

Téma	Kreslící tabulka	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Mnoho z nás si z dětství pamatuje ruční kreslicí tabulku s displejem a dvěma ovládajcími kolečky pro kreslení. Jedno kolečko pro kreslení v horizontálním a druhé ve vertikálním směru. Jelikož tyto hračky nahradily jiné, mnohem propracovanější, setkáme se s takovou tabulkou jen zřídka. Není pro nás ovšem žádný problém si takovou tabulku pomocí robotické stavebnice vyrobit přímo pomocí řídicí jednotky. Jako ovládací prvky nám postačí její tlačítka.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je vytvořit kreslicí tabulku pomocí displeje řídicí jednotky. Při stisku směrových tlačítek se bude vždy vykreslovat do požadovaného směru. Středové tlačítko slouží k mazání plochy displeje.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí vykreslovat či vypisovat různé prvky na plochu displeje řídicí jednotky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s tlačítky řídicí jednotky.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
40 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Vytvořte funkční model kreslicí tabulky. Požadavky na tvorbu budou následující:

1. Pro kreslení využijte plochu displeje řídicí jednotky.
2. Ovládání při vykreslování budou obstarávat tlačítka řídicí jednotky.
3. Prostřední tlačítko bude sloužit k mazání plochy displeje.

Výsledný program ke stažení ve formátu .ev3

Program ke stažení v on-line kurzu.

7 Základní vstupní moduly

Kapitola se věnuje základním vstupním modulům, které obsahuje základní sada stavebnice LEGO Mindstorms EV3. Vstupní zařízení snímají různé hodnoty z okolního prostředí nebo umožňují zasílat podněty zadávané uživatelem. Ty jsou poté na základě programu zpracovány.

V této kapitole naleznete informace o následujících vstupních zařízeních:

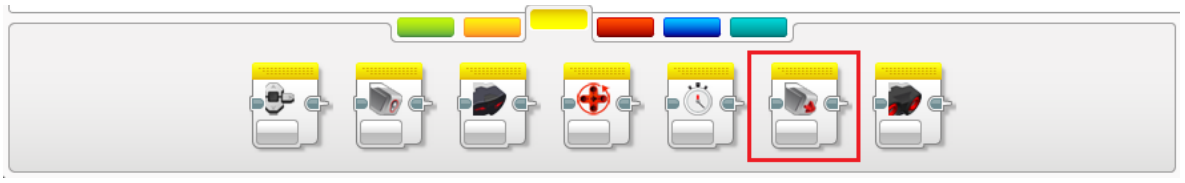
- Dotykový senzor
- Barevný senzor
- Gyroskopický senzor
- Ultrazvukový senzor
- Tlačítka řídicí jednotky

7.1 Dotykový senzor

Možnosti použití dotykového senzoru

Umístění

Blok Touch Sensor pro ovládání dotykového senzoru je umístěn ve žluté záložce Sensor (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy dotykového senzoru

Nastavení portu řídicí jednotky (1-4), ke kterému je senzor připojen, vybereme v malém okně v pravém horním

rohu programového bloku **Měření stavu (Measure State)**

Pokud použijeme senzor v tomto režimu programového bloku, snímá pouze, zda bylo stisknuto tlačítko, či nikoliv. Blok proto obsahuje pouze jediný výstupní port, který nabývá hodnoty logického datového typu (výstupem může být True nebo False). Blok praocující v tomto režimu můžete vidět na obrázku.

Porovnávání stavů (Compare State)

Při přepnutí programového bloku do režimu porovnávání stavů (Compare State) se nám zpřístupní jeden vstupní a dva výstupní

porty bloku. Vstupní port status (State) umožňuje rozlišovat následující tři stavy tlačítka dotykového senzoru:

Released (0) - uvolnění tlačítka
senzoru, Pressed (1) - stisk
tlačítka senzoru,
Bumped (2) - stisk a opětovné uvolnění tlačítka senzoru.

První z výstupních portů (Compare Result) vyjadřuje výsledek porovnání. Jeho hodnota je logického datového typu (True nebo False). Výsledek záleží na tom, zda byl detekován testovaný status tlačítka senzoru.

Druhý výstupní port (Measured Value) vrací číselné označení stavu tlačítka - uvolnění 0, stisk 1 a stisk a opětovné uvolnění 2. Vzhled programového bloku při práci v režimu porovnávání stavů můžete vidět na obrázku.

7.2 Aktivita 7 - Ruční mixer

Téma	Ruční mixér s použitím dotykového senzoru	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Dotykový senzor se hodí pro několik možností využití. Typicky se využívá jako tlačítko. Jeho velmi užitečnou vlastností je možnost rozeznávat tři stavy stisku. Tato aktivita se věnuje jeho použití jako spouštěče určité činnosti jiného zařízení.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem této aktivity je seznámit studenty s možnostmi použití dotykového senzoru. Ten je v úkolu použit jako tlačítko ručního mixéru. Mixér je poháněn pomocí středního motoru. Při ovládání rozlišuje tři stavy. Po prvním stisku a uvolnění se roztočí pomaleji, po dalším stisku maximální rychlostí a při třetím stisku se vypne.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Studenti se naučí využívat dotykový senzor a rozeznávat jeho možné stavy.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Aktivita navazuje na kapitolu 5 Základní orientace v programovacím prostředí EV3.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Stavba modelu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Tvorba programu a průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Závěrečné testování funkčnosti.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na úloze, spolupráce studentů ve skupině a kvalita výsledného modelu a programu.	

Zadání

Aktivita si klade za cíl vás seznámit s funkcemi dotykového senzoru. V následující úloze se naučíte využít dotykový senzor jako spouštěč či přepínač určité reakce. Váš úkol je následující:

1. Postavte vhodný model ručního mixéru, který bude ovládaný pomocí tlačítka dotykového senzoru.
2. Po prvním stisknutí se mixér roztočí pomaleji, při druhém zrychlí na maximum a při třetím zastaví.

Výsledný program ke stažení ve formátu .ev3

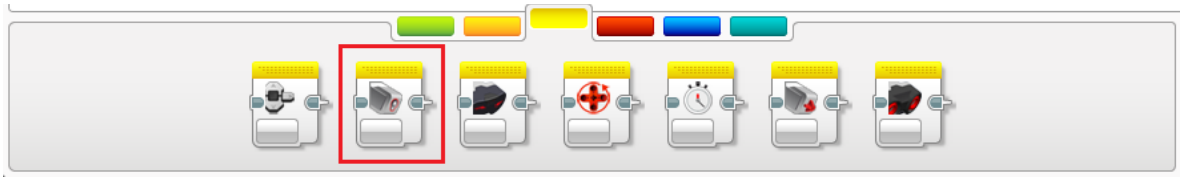
Program ke stažení v on-line kurz.

7.3 Barevný senzor

Možnosti použití dotykového senzoru

Umístění

Blok Color Sensor pro ovládání dotykového senzoru je umístěn ve žluté záložce Sensor (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.

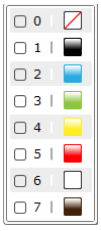


Režimy barevného senzoru

Nastavení portu řídicí jednotky (1-4), ke kterému je senzor připojen, vybereme v malém okně v pravém horním rohu programového bloku.

Rozlišení barev (Measure Color)

Režim rozlišení barev umožňuje zjišťovat barvu snímaného předmětu. Výstupem výstupního portu je číslo od 0 do 7 vyjadřující barvy tak, jak můžete vidět na obrázku.



Intenzita odráženého světla (Reflected Light Intensity)

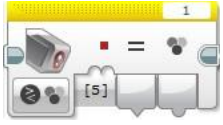
V tomto režimu barevného senzoru můžeme měřit intenzitu odražené světla povrchů rozdílných barev. Každá barva či povrch odráží světlo jinak. Díky výstupnímu portu bloku v tomto režimu získáme intezitu odraženého světla vyjádřenou číselně v rozmezí 0 - 100.

Intenzita okolního světla (Ambient Light Intensity)

Třetím režimem, ve kterém může barevný senzor pracovat, je režim měření intenzity okolního světla. Hodnotou, kterou v tomto režimu výstupní port vrací, je číselná hodnota v rozmezí 0 - 100.

Porovnání barev (Compare Color)

V tomto režimu porovnáváme fixně zadané odstíny barev s barvou, kterou aktuálně senzor detekoval. Na vstupu Set of colors zvolíme barvy, které chceme detekovat. Následují dva výstupní porty. První z nich (Compare Result) je logického datového typu. Jeho výstup bude nabývat hodnoty True v případě, že bude detekována některá ze zvolených barev. V opačném případě bude False. Druhý výstupní port vrací číselné označení (0-7) detekované barvy.

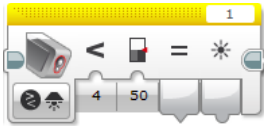


Porovnání intenzity odraženého světla (Compare - Reflected light intensity)

Režim pro porovnání hodnoty intenzity odraženého světla porovnává zjištěnou hodnotu s fixně zadanou prahovou hodnotou (Treshhold value). Režim obsahuje také dva výstupní porty. První s názvem Compare Value vrací hodnotu logického datového typu. Nabývá hodnoty True v případě, že je splněna podmínka porovnání. V opačném případě vrací False. Druhý výstupní port vrací zjištěnou intenzitu světla na stupnici 0 - 100.

Porovnání intenzity okolního světla (Compare - Ambient light intensity)

Programový blok v tomto režimu pracuje naprosto stejně jako v režimu porovnání intenzity odraženého světla. Pouze pracuje s hodnotou intenzity světla v okolí.



Kalibrace - nastavení minima

Při práci s intenzitou odráženého světla je často nutné v programu určit, která hodnota má být vyhodnocována jako nejnižší a která jako nejvyšší. K tomu slouží u barevného senzoru kalibrace. V režimu minimum nastavuje hodnotu, která je dále v programu vyhodnocována jako minimální možná.

Kalibrace - nastavení maxima

Opakem k režimu Minimum je Maximum. V tomto režimu nastavujeme hodnotu, která bude dále v programu vyhodnocována jako maximání možná.

Kalibrace - nulování

Tento režim barevného senzoru nastaví znovu senzor do základního nastavení.

7.4 Aktivita 8 - Rozpoznávač barev

Téma	Rozpoznávač barev s využitím barevného senzoru	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Ne každý z nás má barevné cítění a okamžitě rozpozná, o jakou barvu se jedná. Pociťujeme to hlavně při nákupu oblečení. Barev existuje tolik, že se v nich laik téměř nevyzná. Díky robotické stavebnici si ale dokážeme sestrojit jednoduchý rozpoznávač barev, který nám alespoň základní barvy dokáže určit.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem studentů je vytvořit jednoduchý model ručního rozpoznávače barev, který bude opatřen barevným senzorem. Ten bude zjišťovat barevný odstín snímaného materiálu a barvu následně vypíše na displej řídicí jednotky.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Studenti se naučí pracovat s barevným senzorem, seznámí se s jeho možnými režimy a s jeho využitím sestrojí jednoduché zařízení k rozpoznávání barev.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Aktivita navazuje na kapitolu 6 Základní výstupní moduly.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace), fyzika (světlo).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Stavba modelu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Tvorba programu a průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Závěrečné testování a praktické ověření funkčnosti.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a kvalita a funkčnost sestaveného modelu a vytvořeného programu.	

Zadání

V této aktivitě je vaším úkolem sestavit jednoduchý ruční rozpoznávač barev. Pro úspěšné splnění úlohy naplňte následující požadavky:

1. Sestavte model ručního rozpoznávače barev, který bude možné držet v jedné ruce (využijte barevný senzor).
2. Jakmile namíříte senzor na povrch, jehož barvu potřebujeme zjistit, informace o zjištěné barvě se vypíše na displej řídicí jednotky.

Výsledný program ke stažení ve formátu .ev3

Program ke stažení v on-line kurzu.

7.5 Gyroskopický senzor

Možnosti použití gyroskopického senzoru

Umístění

Blok Gyro Sensor pro ovládání ultrazvukového senzoru není součástí základní instalace Home verze programovacího prostředí EV3. Do programovacího prostředí je nutné jej doinstalovat. Blok naleznete ke stažení na stránkách společnosti LEGO (viz. on-line kurz)

Po instalaci se blok umístí do žluté záložky Sensor (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.

Režimy gyroskopického senzoru

Nastavení portu řídicí jednotky (1-4), ke kterému je senzor připojen, vybereme v malém okně v

pravém horním rohu programového bloku.

Měření úhlu natočení ve stupních (Measure Angle)

Režim měření úhlu umožňuje získávat údaje o míře natočení modelu robota, na kterém je gyroskopický senzor umístěn, v úhlových stupních. Jediný výstupní port vrací tento údaj v číselném vyjádření.

Rychlost rotace ve stupních za sekundu (Measure Rate)

Druhý režim měření s pomocí gyroskopického senzoru umožňuje získávat pomocí jediného výstupního portu hodnotu rychlosti rotace robota ve stupních za sekundu.

Měření úhlu natočení a rotace ve stupních za sekundu (Measure Angle and Rate)

Režim je kombinací obou předchozích.

Porování úhlu natočení ve stupních (Compare Angle)

V tomto režimu porovnání můžeme poměřovat naměřenou hodnotu natočení s fixně zadanou hodnotou. Návratovou hodnotou je nám buďto výsledek splnění této podmínky (Compare Result) v podobě hodnoty True (při splnění), nebo False (při nesplnění). Druhou možností je vrácená hodnota udávaná ve stupních, kterou získáme pomocí druhého výstupního portu (Angle).

Porovnání naměřené rotace ve stupních za sekundu (Compare Rate)

Režim umožňuje totožné možnosti práce s hodnotou naměřenou gyroskopickým senzorem jako předchozí režim (Compare Angle). Místo úhlu natočení ve stupních ale pracuje s hodnotou naměřené rotace ve stupních za sekundu.

Nulování naměřeného natočení (Reset)

Režim Reset slouží k nulování posledního naměřeného natočení. Měření úhlu natočení se totiž určuje oproti poslednímu stavu robota. Je proto dobré vždy před měřením naměřenou hodnotu nulovat touto funkcí.

7.6 Aktivita 9 - Detektor pádu pevného disku

Téma	Detektor pádu disku	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Nenadálé problémy se spuštěním počítače mohou někdy pramenit z poruchy pevného disku počítače. Stát se tak může i v jiných případech. Například při pádu spuštěného notebooku na zem. Ať již je příčina jakákoliv, ve velké většině případů nastane nepříjemná ztráta dat. Pevné disky ovšem obsahují zařízení, které při detekci pádu dokáže bezpečně zaparkovat zápisovou hlavu tak, aby nedošlo k poškození disku. My si takové zařízení zkusíme vytvořit díky robotické stavebnici za pomoci gyroskopického senzoru.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve sestaví jednoduchý model pevného disku. Uvnitř se pokusí vytvořit pohyblivý mechanismus, který bude simulovat jeho funkci (např. motor nebo pohyblivé rameno). Zařízení pro detekci pádu bude představovat gyroskopický senzor. Následně vytvoří program, který při prudké změně polohy zastaví chod disku.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat gyroskopický senzor.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s barevným senzorem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu disku.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Testování funkčnosti modelu a	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujících multimediálních prvků.	

Zadání

Vytvořte funkční model pevného disku se zařízením pro detekci pádu. Požadavky na konstrukci modelu a program budou následující:

1. Sestavte jednoduchý model pevného disku.
2. Uvnitř bude umístěn pohyblivý mechanismus poháněný například motorem.
3. Pro detekci pádu použijte gyroskopický senzor.
4. Jakmile bude detekována prudká změna polohy, chod disku se zastaví.

Výsledný program ke stažení ve formátu .ev3

Program ke stažení viz. on-line kurz

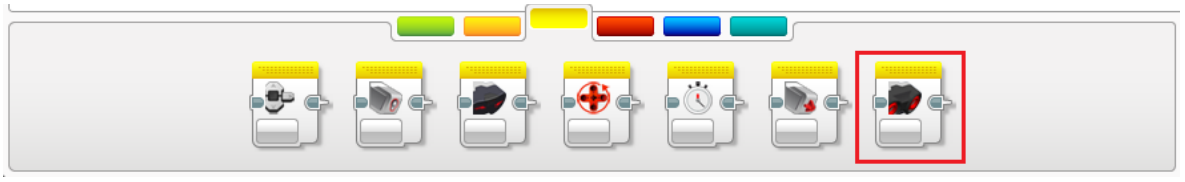
7.7 Ultrazvukový senzor

Možnosti použití ultrazvukového senzoru

Umístění

Blok Ultrasonic Sensor pro ovládání ultrazvukového senzoru není součástí základní instalace Home verze programovacího prostředí EV3. Do programovacího prostředí je nutné jej doinstalovat. Blok naleznete ke stažení na stránkách společnosti LEGO (viz. on-line kurz).

Po instalaci se blok umístí do žluté záložky Sensor (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



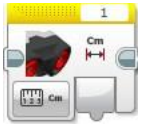
Režimy ultrazvukového senzoru

Nastavení portu řídicí jednotky (1-4), ke kterému je senzor připojen, vybereme v malém okně v pravém horním

rohu programového bloku.

Měření vzdánosti - v centimetrech nebo palcích (Measure Distance - Centimeters, Inches)

První dva režimy, ve kterých může ultrazvukový senzor pracovat, slouží k měření vzdálenosti. První vyjadřuje naměřenou vzdálenost v centimetrech a druhý v palcích. V obou režimech se u programového bloku zobrazuje pouze jediný výstupní port, který vrací naměřenou hodnotu.



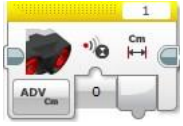
Detekce ultrazvukového signálu (Measure Presence)

Režim detekce ultrazvukového signálu pouze ověřuje, zda senzor zachytil nějaký ultrazvukový signál, či nikoliv. Blok v tomto režimu obsahuje pouze jediný výstupní port, který je logického datového typu. Hodnotu True vrací v případě, že signál zachytí. Ve všechno ostatních případech vrací False.

Rozšířené měření vzdálenosti (Measure Advanced - Centimeters, Inches)

Kromě jednoduchého režimu měření vzdálenosti obsahuje progmamový blok pro ultrazvukový senzor také rozšířený režim. Od původního režimu se liší možností nastavení způsobu vysílání ultrazvukového signálu. K dispozici máme tyto dvě možnosti:

Ping (návratová hodnota 0) - vysílání signálu v pravidelných intervalech.
Continuous (návratový hodnota 1) - souvislé (neustálé) vysílání signálu bez přerušení.



Porovnání vzdáleností v centimetrech nebo palcích (Compare Distance - Centimeters, Inches)

Režim slouží k porovnání vzdálenosti snímané senzorem s fixně zadanou hodnotou. Výstupem programového bloku v tomto režimu může být výsledek porovnání (výstupní port Compare Result), který vyjadřuje, zda byla splněna ověřovaná podmínka. Výstup je logického datového typu. Hodnotu True vrací v případě splnění podmínky, v opačném případě vrací False. Druhým výstupním portem je vzdálenost snímaná senzorem vyjádřená v centimetrech nebo v palcích.

Detekce ultrazvukového signálu (Compare Presence/Listen)

Režim, který je totožný s režimem Measure Presence.

7.8 Aktivita 10 - Turniket

Téma	Inteligentní turniket	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Na hudebních koncertech a různých sportovních akcích se zjišťuje různými způsoby aktuální počet návštěvníků. Většinou se tomu tak děje díky tuniketům, které sčítají počet lidí, kteří jimi projdou. V následující úloze si takový turniket vytvoříme za pomoci ultrazvukového senzoru.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	V této úloze studenti vytvoří model jednoduchého turniketu, který sčítá počet návštěvníků kulturní akce. Ultrazvukový senzor bude přesně snímat prostor určení pro příchod do areálu. Jakmile jeho vysílaný signál protne některý z příchozích návštěvníků, bude započítán. Počet návštěvníků se bude postupně přičítat a vypisovat na displej.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Studenti se naučí pracovat s ultrazvukovým senzorem a naučí se zpracovávat jím zjištěná data.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Aktivita navazuje na kapitolu 5 Základní orientace v programovacím prostředí EV3.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Tvorba modelu turniketu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
25 minut	Tvorba programu a průběžné	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Ověření funkčnosti a závěrečné	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a kvalita sestaveného modelu a funkčnost vytvořeného programu.	

Zadání

V této aktivitě se seznámíte s funkcí ultrazvukového senzoru. Vaším úkolem je vytvořit inteligentní turniket pro kulturní akce, který sčítá počet návštěvníků. Požadavky na jeho funkčnost jsou následující:

1. Vytvořený model musí být jednoduchý a plně funkční (využijte ultrazvukový senzor).
2. Průchod turniketem musí být snímán ultrazvukovým senzorem, který bude detekovat každého příchozího návštěvníka.
3. Jakmile návštěvník turniketem projde, bude zaznamenán.
4. Aktuální počet návštěvníků se bude vypisovat na displeji řídicí jednotky.

Výsledný program ke stažení ve formátu .ev3

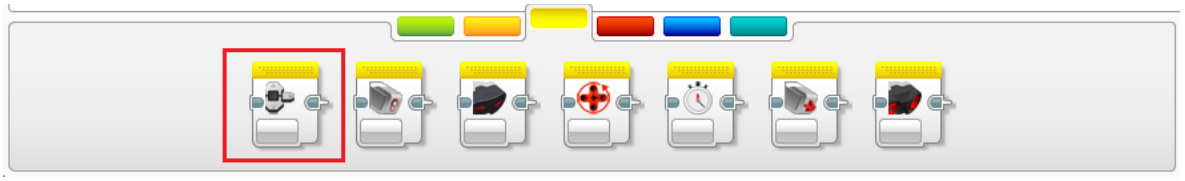
Program ke stažení viz. on-line kurz

7.9 Tlačítka řídicí jednotky

Možnosti použití tlačítek řídicí jednotky

Umístění

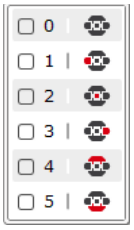
Blok Brick Buttons pro řízení reakcí na stisk tlačítek řídicí jednotky je umístěn ve žluté záložce Sensor (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy pro reakci na stisk tlačítek řídicí jednotky

Detekce stisku tlačítka (Measure Brick Buttons)

První režim, ve kterém můžeme pomocí bloku Brick Buttons pracovat se stiskem tlačítek řídicí jednotky, je režim pro detekci jejich stisknu. Pomocí výstupního portu, který vrací číselné označení tlačítka (0 - 5), můžeme získat informaci o tom, které tlačítko řídicí jednotky bylo stisknuto. Schéma číslování tlačítek (návratové hodnoty) můžete vidět na obrázku.



Compare Brick Buttons

Režim pro porovnání stisknutých tlačítek umožňuje zjišťovat, zda bylo stisknuto tlačítko, které požadujeme. Na prvním vstupním portu si zvolíme tlačítko, které chceme testovat. Pomocí druhého vstupního portu můžeme dokonce testovat, zda bylo stisknuto, uvolněno nebo stisknuto i uvolněno. První výstupní port Compare Result vrací návratovou hodnotu logického datového typu. Ta nabývá hodnoty True v případě, že je stisknuto testované tlačítko. V opačném případě vrací False. Druhý výstupní port vrací číselné označení stisknutého tlačítka (0 - 5).

7.10 Aktivita 11 - Bomba

Téma	Bomba	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Práce pyrotechnika není záviděníhodná. Stojí za ní roky zkušeností, znalostí elektrotechniky a hlavně pevné nervy. Zkušený pyrotechnik ovšem dokáže vyhodnotit povahu výbušniny a její funkci a zneškodnit ji. Vaším úkolem v této aktivitě ale bude vytvořit nepředvídatelnou bombu, jejíž chování nebude možné zjistit. Bude záviset pouze na štěstí, zda se ji povede zneškodnit, či nikoliv.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. a 2. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je vytvořit program, který bude simulovat bombu. Potenciálnímu pyrotechnikovi se na displeji zobrazí výzva, aby bombu zneškodnil jedním ze tří nabízených tlačítek. U žádného z nich ovšem nebude zaručeno, že je správné. Význam tlačítek se totiž bude náhodně generovat. Program žáci doplní o vhodnou zvukovou signalizaci a výpis na displej.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Studenti se naučí pracovat s tlačítky řídicí jednotky a naučí se vyhodnocovat jejich stisk a reagovat na něj.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s tlačítky řídící jednotky.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
40 minut	Tvorba programu a jeho průběžné	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a funkčnost vytvořeného programu.	

Zadání

V této aktivitě se naučíte využívat tlačítka řídící jednotky. Vaším úkolem je vytvořit z řídicí jednotky pomyslný model bomby. Co pro úspěšnou realizaci musíte zvládnout?

1. Zvolte si alespoň tři tlačítka řídicí jednotky, která budou představovat dráty, které musí pyrotechnik přestříhnout.
2. Programově zajistěte, aby žádné tlačítko nebylo určeno ke zneškodnění bomby, ale aby se jejich funkce náhodně měnila.
3. Program vhodně graficky a zvukově ošetřete.

Výsledný program ke stažení ve formátu .ev3

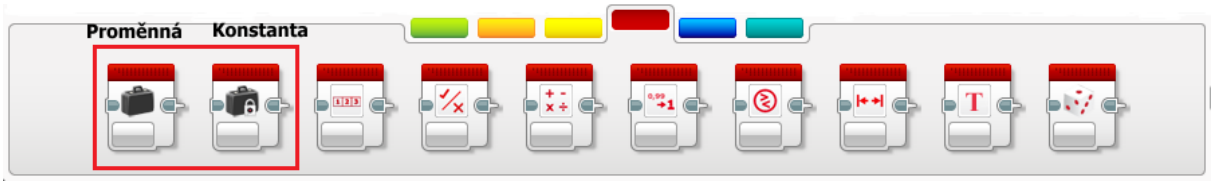
Program ke stažení viz. on-line kurz

8 Proměnné, konstanty, datové typy

Možnosti využití proměnné

Umístění

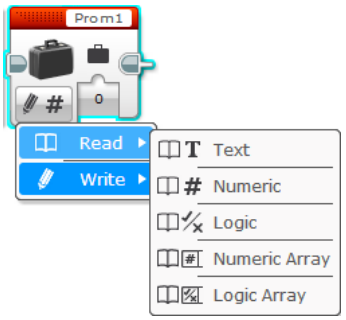
Blok Variable pro použití proměnné a blok Constant pro použití konstanty jsou umístěny v červené záložce Data Operations. Použít je můžeme přetažením na programovací plochu. Rozeznáme je lišícím se vzhledem programového bloku. Konstanta obsahuje malý symbol zámku značící fixní hodnotu, která je v ní uložená.



Možnosti čtení a zápisu

Konstantu je možné použít pouze pro čtení hodnoty v ní uložené. Proměnnou je možné použít jak pro čtení dat v ní uložených, tak pro zápis. U čtení i u zápisu musíme zvolit datový typ, kterého bude proměnná nabývat.

Při programování v EV3 můžeme využít následující datové typy: Text - pro textové a znakové řetězce,
Numeric - pro čísla,
Logic - pro hodnoty logického datového typu (True a False), Numeric Array - pro číselná pole,
Logic Array - pro pole obsahující hodnoty logického datového typu.



Rozdíl mezi proměnnou pro čtení a pro zápis

Proměnná pro čtení umožňuje využít uloženou hodnotu. Pomocí datového vodiče ji proto můžeme z výstupního portu bloku přivést na vstupní port jiného programového bloku. U proměnné pro zápis můžeme naopak hodnotu do bloku Variable přivést, a tedy uložit. Rozdíl mezi blokem nastaveným pro čtení a pro zápis můžete vidět na obrázku.

Proměnná pro čtení **Proměnná pro zápis**

9 Programové řízení

V programování využíváme často konstrukce, které se v programu vykonávají opakovaně. Abychom je nemuseli v kódu zapisovat několikrát, což by mohlo být v některých případech nereálné, využíváme cykly. V případech, kdy potřebujeme rozhodnout o naplnění nějakého stavu, hodnoty nebo adekvátně reagovat na nastalé podmínky, využijeme podmíněné výrazy. Tato kapitola se věnuje možnostem využití cyklů a podmínek v programovacím prostředí EV3.

Podkapitoly:

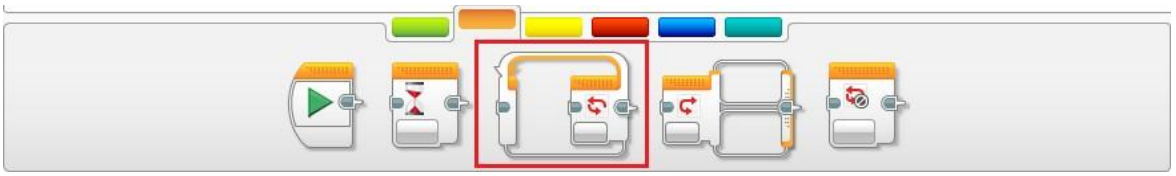
- Cykly
- Podmínky

9.1 Cykly

Možnosti použití cyklu v EV3

Umístění

Blok Loop, který v EV3 reprezentuje cyklus, je umístěn v oranžové záložce Flow Control (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Popis bloku cyklu

Název cyklu - v horní části cyklu je možné volit jeho název (v políčku, ve kterém je defaultně nastaveno číslování od 01 dále). Volba režimu - slouží ke zvolení způsobu řízení cyklu (například počtem průchodů, časem, senzorem atd.).

Výstupní port - vyjadřuje počet opakování cyklu.

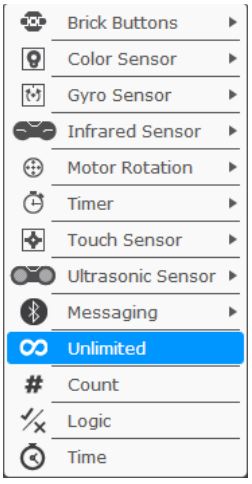
Vstupní port - závisí na zvoleném režimu (ovlivňuje například počet průchodů cyklem - u režimu Count). Prostor pro umístění programových bloků - slouží k umístění bloků programu, které se budou vykonávat.

Možnosti řízení cyklu

Programovací prostředí EV3 neumožňuje vytvářet cykly s podmínkou na začátku, ale pouze s podmínkou na konci a s pevným počtem průchodů. Na obrázku můžete vidět možnosti řízení cyklu:

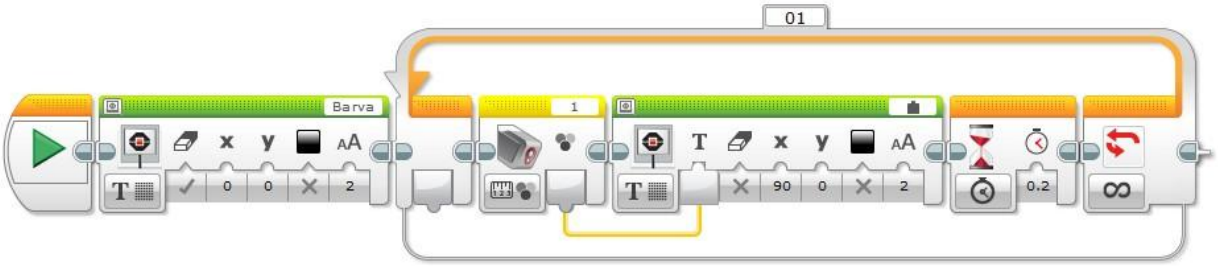
Senzorem - počet možností řízení senzorem se odvíjí od počtu nainstalovaných programových bloků pro ovládání senzorů. Nekonečné provádění cyklu (Unlimited) - možnost neustálého provádění (nekonečný cyklus).
Cyklus s pevným počtem průchodů (Count) - cyklus je vykonáván na základě nastaveného počtu průchodů.

Cyklus řízený logickou hodnotou (Logic) - cyklus vykonávaný, dokud je naplněna logická hodnota True nebo False. Cyklus řízený časem (Time) - cyklus je vykonáván po určitou dobu uváděnou v sekundách.



Příklad použití cyklu

Na obrázku můžete vidět příklad použití cyklu. Jednoduchý programový konstrukt díky nekonečnému provádění cyklu neustále zjišťuje, jaké barvy je snímaný povrch pomocí barevného senzoru. Hodnotu poté vypisuje na displej.

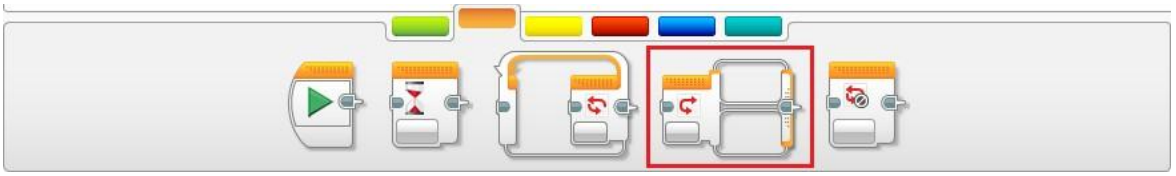


9.2 Podmínky

Možnosti použití podmínek v EV3

Umístění

Blok Switch, který v EV3 reprezentuje podmínku, je umístěn v oranžové záložce Flow Control (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.

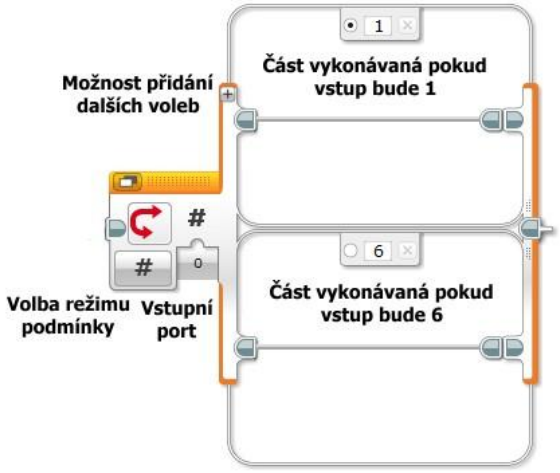


Popis bloku cyklu

Volba režimu - slouží ke zvolení způsobu řízení podmínky (například vstupní číselnou hodnotou, logickou hodnotou, senzorem atd.). Vstupní port - slouží k testování porovnávané hodnoty s hodnotou fixně zadanou, testování specifických hodnot snímaných senzorem atd.).

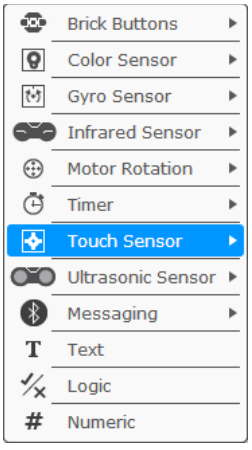
Prostor pro umístění programových bloků - v základu máme k dispozici část, do které umisťujeme sekvenci programových bloků, která se vykoná v případě, že je podmínka splněna, a druhou část, která se vykoná, když podmínka splněna není. Blok switch umožňuje vytvořit také přepínač case. Slouží k tomu malá ikonka plus v horní části bloku, díky které můžeme přidat další možnosti. Na základě toho můžeme například u barevného senzoru rozlišovat, která barva byla detekována, a na základě toho programově reagovat (podobu bloku Switch při práci s barevným senzorem vidíte na obrázku).

Volba portu - zobrazuje se (v levé části bloku) pouze v případě, že je podmínka řízena senzorem.



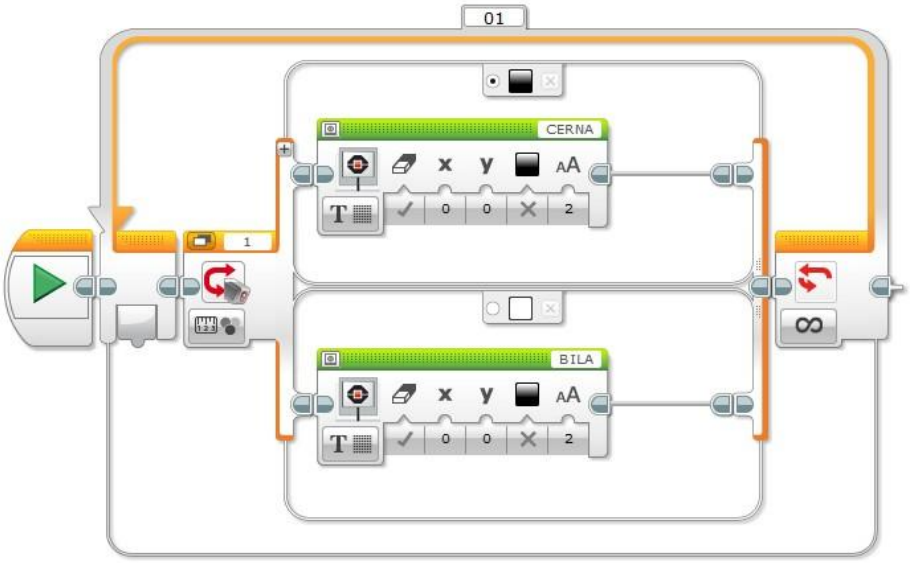
Možnosti řízení podmínky

Programovací prostředí EV3 neumožňuje řídit podmínky několika způsoby, které můžete vidět na obrázku: Senzorem - vykonávání podmínky je řízeno vstupní hodnotou zjištěnou senzorem.
Řetězcem znaků - podmínka je vykonávána na základě ověřování textového řetězce přivedeného na vstup. Logic - vykonávání podmínky na základě ověřování logické hodnoty (True nebo False) na jejím vstupu. Numeric - podmínka vykonávaná na základě porovnání číselné hodnoty na vstupu.



Příklad použití cyklu

Na obrázku můžete vidět příklad použití kombinace podmínky a cyklu. Program ověřuje, zda byla detakována černá nebo bílá barva pomocí barevného senzoru, a výsledek vypisuje na displej.



9.3 Aktivita 12 - Detektor světla

Téma	Detektor světla	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Zařízení detekující úroveň světla v okolí nalezneme v mnoha mechanismech. Jedním z nich může být například automatické osvětlení, které na základě detekce světelných podmínek spouští pouliční osvětlení, či nikoliv. My se v této aktivitě pokusíme takový detektor sestrojit. Využijeme k tomu barevný senzor, který režim detekce světla v okolí obsahuje.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SS a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je sestrojit a naprogramovat jednoduchý detektor světla, který rozlišuje světelné podmínky v okolí. Na displeji následně zobrazuje, zda je světla v okolí dostatek, či nikoliv.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat cykly a podmínky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s cykly a podmínky. Znalost režimů barevného senzoru.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Casový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Tvorba programu a jeho průběžné	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a funkčnost vytvořeného programu.	

Zadání

V této aktivitě se naučíte využívat tlačítka řídicí jednotky. Vaším úkolem je vytvořit z řídicí jednotky pomyslný model bomby. Co pro úspěšnou realizaci musíte zvládnout?

1. Zvolte si alespoň tři tlačítka řídicí jednotky, která budou představovat dráty, které musí pyrotechnik přestříhnout.
2. Programově zajistěte, aby žádné tlačítko nebylo určeno ke zneškodnění bomby, ale aby se jejich funkce náhodně měnila.
3. Program vhodně graficky a zvukově ošetřete.

Výsledný program ke stažení ve formátu .ev3

Program ke stažení viz. on-line kurz

10 Matematické operace

Provádění matematických operací je nezbytnou nutností v programování v jakémkoliv jazyce. V této kapitole se dozvíte o následujících matematických operacích:

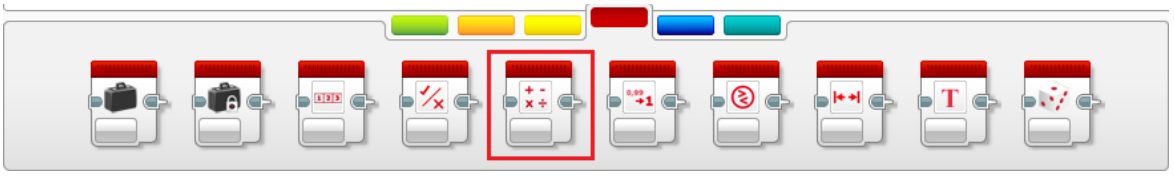
- Základní matematické operace
- Logické operace
- Pole
- Generování náhodných čísel
- Rozšiřující matematické operace

10.1 Základní matematické operace

Možnosti využití základních matematických operací

Umístění

Blok Math pro provádění základních matematických operací je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy pro provádění základních metamatických operací

Sčítání (Add)

Režim pro sčítání (Add) umožňuje sečíst dvě vstupní hodnoty. Ty můžeme na portech A a B buďto ručně zadat, nebo je přivést pomocí datového vodiče. Výsledek poté získáme pomocí výstupního portu Result.

Odčítání (Subtract)

Režim pro odčítání (Subtract) umožňuje odečíst dvě vstupní hodnoty. Můžeme je buďto ručně zadat na portech A a B, nebo přivést pomocí datového vodiče. Výsledek opět získáme na výstupním portu Result.

Dělení (Divide)

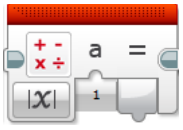
Režim pro dělení funguje principiálně stejně jako režimy pro sčítání a odčítání. Na vstupních portech A a B zadáme vstupní hodnoty a na výstupu Result následně získáme výsledek.

Násobení (Multiply)

Režim pro násobení je opět totožný s předchozími. Na vstupních portech A a B zadáme vstupní hodnoty a na výstupu Result poté získáme výsledek.

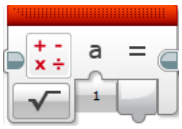
Absolutní hodnota (Absolut Value)

Režim Absolut Value vrací na výstupním portu Result absolutní hodnotu čísla zadaného na vstupním portu A.



Odmocnina (Square Root)

Režim Square Root vrací na výstupním portu Result odmocninu čísla zadaného na vstupním portu A.



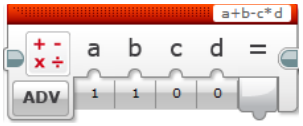
Mocnina (Exponent)

Režim Exponent umožňuje pracovat s odmocninami. Na vstupním portu A zadáme základ mocniny (číslo) a na vstupním portu N požadovaný exponent. Výsledek operace následně vrací výstupní port Result.

Advanced (Pokročilé funkce)

V režimu Advanced je možné počítat složitější matematické výrazy či rovnice. K dispozici máme 4 vstupní porty (A, B, C, D) pro zadání číselných hodnot. Výsledek následně vrací výstupní port Result.

Zápis rovnice či výrazu se provádí v okně, které se zobrazí po kliknutí do bílého pole v pravém horním rohu bloku. K zápisu lze využít množství operací a funkcí (základní matematické operace, goniometrické funkce, mocniny, odmocniny a další).

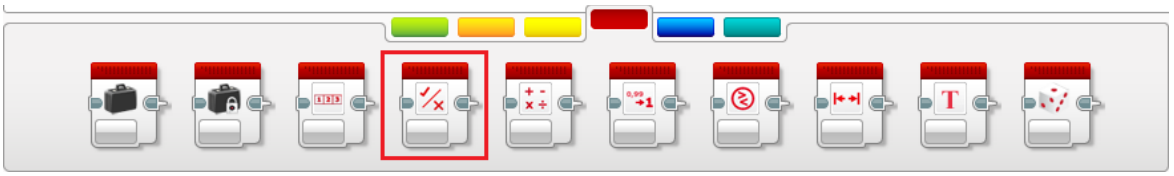


10.2 Logické operace

Možnosti využití logických matematických operací

Umístění

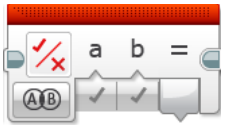
Blok Logic Operations pro provádění logických matematických operací je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy pro provádění logických metamatických operací

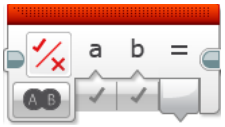
Logický součin (And)

Režim pro logický součin obsahuje dva vstupní porty logického datového typu (mohou nabývat hodnot True a False). Výsledek operace je True v případě, že hodnoty obou vstupních portů nabývají hodnotu True. Výsledek vrací výstupní port Result.



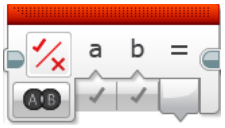
Logický součet (Or)

Režim pro logický součet obsahuje dva vstupní porty A a B, které jsou logického datového typu (nabývají hodnot True nebo False). Výsledem operace, který vrací výstupní port Result, je True v případě, že hodnota alespoň jednoho vstupního portu je True.



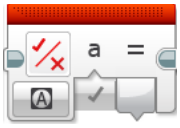
Exkluziv Or (XOR)

Režim exkluziv Or obsahuje, stejně jako oba předchozí režimy, dva vstupní porty (A a B) logického datového typu. Výsledek vrací výstupní port Result. Výsledkem operace je True v případě, že pouze jeden ze vstupních portů nabývá hodnoty True.



Negace (Not)

Režim Not umožňuje negaci vstupní hodnoty logického datového typu na portu A. Výsledkem je tedy obrácená hodnota.

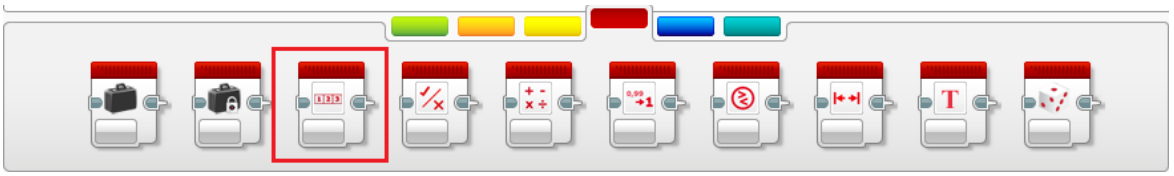


10.3 Pole

Možnosti práce s polem

Umístění

Blok Array Operations pro práci s polem je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy pro práci s polem

Připsání číselné hodnoty (Append- Numeric)

Režim Append s volbou Numeric slouží k přidání číselné položky na konec pole. Pomocí tohoto režimu je také možné vytvořit pole nové. Na vstupním portu Array In (vstup) se zadávají položky nově vytvářeného pole. Pokud uživatel nezadá nic, do pole se zapíše pouze nově zadávaná položka. Ta se zadává na vstupním portu Value. Výstupní port Array Out (výstup pole) vrací výsledek celé operace (pole s přidanou položkou).

Připsání logické hodnoty (Append - Logic)

Princip fungování režimu je naprosto totožný s režimem Append - Numeric s tím rozdílem, že se nejedná o číselné pole, ale o pole logických hodnot.

Přečtení prvku pole (Read at Index - Numeric)

V tomto režimu vrací programový blok na výstupním portu Value hodnotu zvoleného prvku pole. Na vstupním portu Array In si zvolíme pole, ze kterého chceme číst, a na portu Index pořadové číslo (index) požadovaného prvku (první prvek pole má index 0).

Příklad: Pokud zadáme čtení indexu 2 pro pole [1,2,3], vráceným výsledkem na výstupním portu Value bude hodnota 3.

Přečtení prvku pole (Read at Index - Logic)

Režim pro čtení prvku pole logických hodnot je totožný s režimem pro čtení prvku číselného pole. Výslednou hodnotou je tedy hodnota logického datového typu.

Zápis prvku pole na pozici (Write at Index - Numeric)

Režim umožňuje zapsat na zvolenou pozici číselného pole libovolnou číselnou hodnotu. Na vstupním portu Array In zvolíme pole, do kterého se má zapisovat. Na vstupním portu Index pozici, na kterou se má zvolená hodnota, kterou zadáme na vstupním portu Value, zapsat. Výsledem operace vráceným na výstupním portu Array Out je pole s námi zapsaným prvkem.

Příklad: Máme definované pole hodnot [1,2,3,4,5]. Na pozici číslo 3 chceme zapsat číslo 9. Výsledkem operace bude pole [1,2,3,9,5].

Zápis prvku pole na pozici (Write at Index - Logic)

Režim pro zápis prvku pole logických hodnot obsahuje totožné volby jako předchozí režim pro číselná pole. Na zvolenou pozici se ovšem zapisuje hodnota logického datového typu.

Délka pole (Length - Numeric)

Režim Length vrací délku (počet prvků) zvoleného pole. Na vstupním portu Array In si zvolíme číselné pole. Výstupní port Length následně vrací délku zvoleného pole.

Příklad: Pokud chceme pomocí programového bloku pro práci s polem zjistit délku pole [1,2,3,4,5], výsledkem operace bude číslo 5 (pole obsahuje 5 prvků).

Délka pole (Length - Logic)

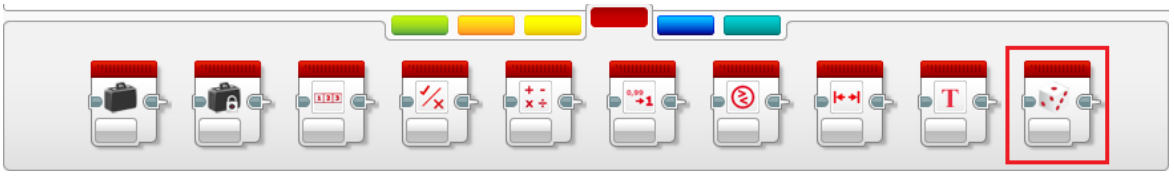
Režim Length pro pole logických hodnot umožňuje stejné možnosti jako režim pro zjišťování délky číselného pole.

10.4 Generování náhodných čísel

Možnosti generování náhodných čísel v EV3

Umístění

Blok Random pro generování náhodných čísel je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy pro generování náhodných čísel

Generování čísel (Numeric)

Režim Numeric umožňuje generovat náhodné číslo ze zvoleného rozpětí hodnot. Na prvním vstupním portu Lower Bound (spodní hranice) zadáváme spodní hranici rozpětí hodnot, ze kterého se má číslo generovat. Na druhém vstupním portu Upper Bound (horní hranice) poté zadáváme horní hranici číselného rozpětí. Výstupní port Value následně vrací náhodně vygenerované číslo ze zvoleného rozpětí.

Generování logických hodnot (Logic)

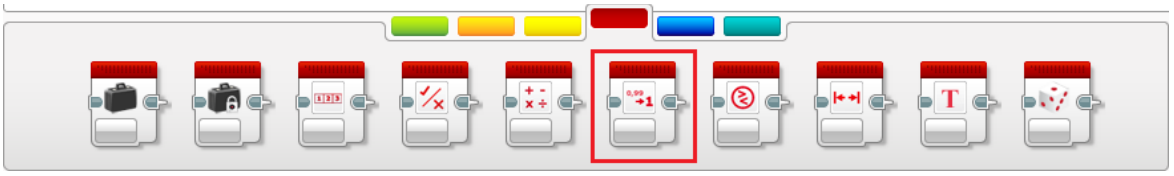
Režim pro generování náhodných hodnot obsahuje pouze jeden vstupní port Probability of True (pravděpodnost vygenerování hodnoty True). Ten nabývá hodnoty od 0 do 100 % a můžeme s jeho pomocí ovlivnit, jaká bude procentuálně vyjádřená pravděpodobnost, že náhodně vygenerovaná logická hodnota bude True. Výsledek generování poté vrací výstupní port Value.

10.5 Další datové operace

Možnosti zaokrouhlování číslených hodnot

Umístění

Blok Round pro zaokrouhlování číselných hodnot je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy bloku pro zaokrouhlování

K nejbližšímu (To Nearest)

Režim umožňuje zaokrouhlovat číselnou hodnotu zadanou na vstupním portu Index k nejbližšímu celému číslu. Zaokrouhlednou hodnotu vrací výstupní port Result.

Příklad: Číslo 4,75 bude zaokrouhleno na 5.

Nahoru (Round Up)

Režim umožňující zaokrouhlování číselných hodnot zadaných na vstupním portu Index směrem nahoru. Zaokrouhlednou hodnotu vrací výstupní port Result. *Příklad:* Číslo 4,75 bude zaokrouhleno na 5.

Dolů (Round Down)

Režim umožňující zaokrouhlovat číslenou hodnotou zadanou na vstupním portu Index směrem dolů. Zaokrouhlednou hodnotu vrací výstupní port Result. *Příklad:* Číslo 4,75 bude zaokrouhleno na 4.

Odříznutí desetinné části (Truncate)

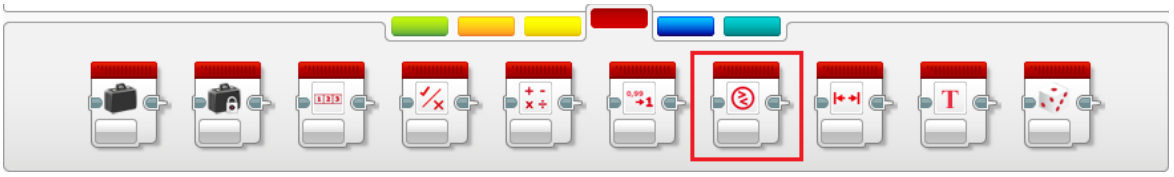
Režim Truncate umožňuje oříznutí čísla na zvolený počet desetinných míst. Do pole Input zadáme desetinné číslo a u možnosti Number of Decimals zvolíme požadovaný počet desetinných míst. Výsledek operace vrací výstupní port Result.

Příklad: Pokud zadáme číslo 4,75 a budeme požadovat jeho oříznutí na 1 desetinné místo, výsledkem bude číslo 4,7.

Možnosti porovnávání číslených hodnot

Umístění

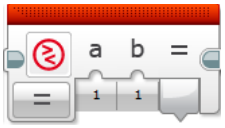
Blok Compare pro porovnávání číselných hodnot je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy bloku pro porovnávání číselných hodnot

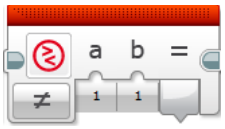
Rovná se (Equal To)

Blok v tomto režimu ověřuje, zda se čísla zadaná na vstupních portech A a B rovnají. Pokud se čísla rovnají, výstupní port Result vrací hodnotu True, v opačném případě False.



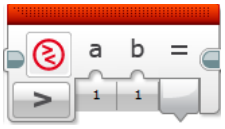
Nerovná se (Not Equal To)

V tomto režimu blok ověřuje, jestli jsou čísla zadaná na vstupních portech A a B rozdílná. Pokud ano, vrací výstupní port Result True, v opačném případě False.



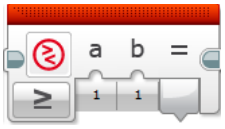
Větší než (Greater Than)

Blok v tomto režimu ověřuje, zda je číslo zadané na vstupním portu A větší než číslo na portu B. Pokud ano, vrací výstupní port Result hodnotu True, v opačném případě False.



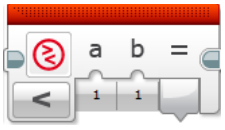
Větší než nebo rovno (Greater Than or Equal To)

Blok v tomto režimu porovnává, zda je číslo zadané na vstupním portu A větší nebo rovno číslu zadanému na portu B. V případě, že ano, vrací výstupní port logickou hodnotu True, v opačném případě False.



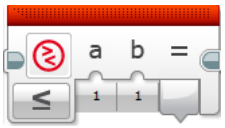
Menší než (Less Than)

Režim, ve kterém programový blok ověřuje, zda je hodnota zadaná na vstupním portu A menší než hodnota na vstupním portu B. Pokud ano, výstupní port Result vrací hodnotu True, v opačném případě False.



Menší než nebo rovno (Less Than or Equal To)

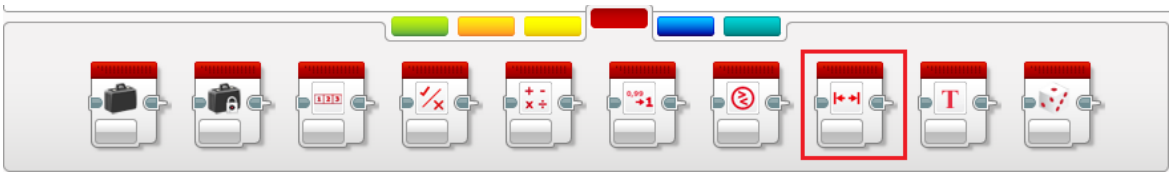
Režim, ve kterém programový blok ověřuje, zda je hodnota zadaná na vstupním portu A menší nebo rovna hodnotě B. Pokud ano, výstupní port Result vrací hodnotu True, v opačném případě False.



Možnosti určování rozmezí čísel

Umístění

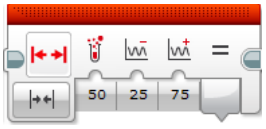
Blok Range pro určování příslušnosti zadaného čísla do rozmezí hodnot je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy bloku pro určování příslušnosti zadaného čísla do rozmezí hodnot

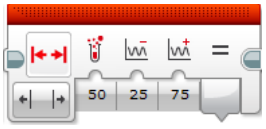
Uvnitř rozmezí (Inside)

Blok v tomto režimu ověřuje, zda je hodnota zadaná na vstupním portu Test Value (testovaná hodnota) uvnitř zadaného rozsahu. Ten se určije na zbylých dvou vstupních portech. Na portu Lower Bound (spodní hranice) zadáváme spodní hranici rozmezí a na vstupním portu Upper Bound (horní hranice) zase horní hranici rozmezí. Výsledek vrácený výstupním portem Result je logického datového typu. Výsledkem je True v případě, že testovaná hodnota je uvnitř zadaného rozsahu, v opačném případě je výsledek False.



Vně rozmezí (Outside)

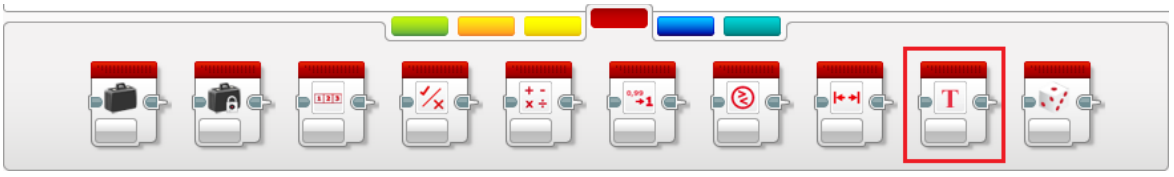
Funkce bloku v tomto režimu je totožná s předchozím režimem. Pouze ověřuje, zda je zadaná hodnota vně určeného rozmezí.



Možnosti spojování textových řetězců

Umístění

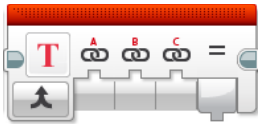
Blok Text pro slučování textových řetězců je umístěn v červené záložce Data Operations (zobrazeno na obrázku). Použít jej můžeme přetažením na programovací plochu a umístěním na požadované místo v programu.



Režimy bloku pro slučování textových řetězců

Spojení (Merge)

Programový blok slouží ke spojení až tří textových retězců (A, B, a C) v jeden. Výsledem vráceným výstupním portem Result jsou tyto zadané řetězce spojené v jeden.



10.6 Aktivita 13 - Stopky

Téma	Stopky	
Tematický celek	Programovací prostředí EV3	
Motivační rámec	Každý správný trenér potřebuje stopky. Využívají je trenéři vrcholových sportovců, závodníků, ale také například učitelé tělesné výchovy. Pomocí nich kontrolují výkony svých svěřenců a ověřují případné zlepšení v jejich sportovných výkonech. Jednoduché stopky si můžeme vytvořit také pomocí robotické stavebnice. Ty naše ovšem budou vhodné spíše pro motoristické sporty či měření běžeckých výkonů.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je sestrojit a naprogramovat jednoduchý model digitálních stopek. Ty budou měřit kolo závodníka, dokud nebude například stisknuto tlačítko dotykového senzoru. Následně uloží časový údaj do pole. Z uložených údajů bude následně vypočítáván průměrný čas na kolo, který bude vypisován na displej.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím EV3.	
Cíle aktivity	Žáci se naučí využívat pole.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Teoretická znalost matematických operací a práce s polem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Casový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Tvorba programu a jeho průběžné	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a funkčnost vytvořeného programu.	

Zadání

Aktivita slouží k tomu, abyste se naučili využívat v programovacím prostředí EV3 pole. Za tímto účelem naprogramujete model stopek, které budou splňovat následující požadavky:

1. Zvolte si, čím budou stopky ovládány (např. tlačítko řídící jednotky nebo tlačítko dotykového senzoru).
2. Stopky se spustí po spuštění programu a po každém stisknutí tlačítka bude zaznamenán čas aktuálně měřeného kola.
3. Ze zaznamenaných časů jednotlivých kol bude vypočítáván průměr, který bude vypisován na displej.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

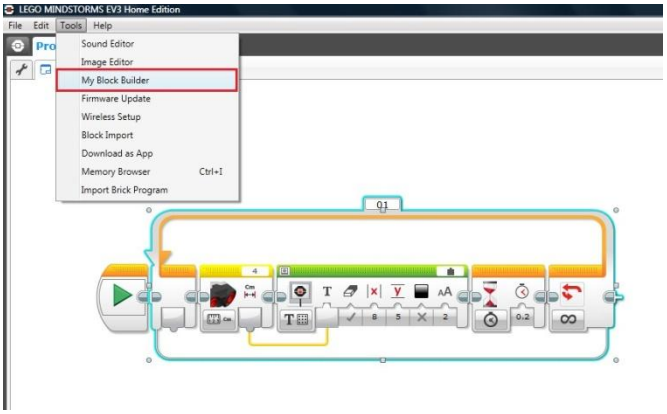
11 Tipy a triky

Vytváření vlastních metod

Vytváření vlastních metod je určeno pro situace, kdy se nám v programu určitý úsek kódu několikrát opakuje, nebo pro případy, kdy si chceme usnadnit práci a zdrojový kód zpřehlednit. Jeho princip spočívá ve vytvoření jediného programového bloku, který obsahuje sekvenci zdrojového kódu (několik bloků).

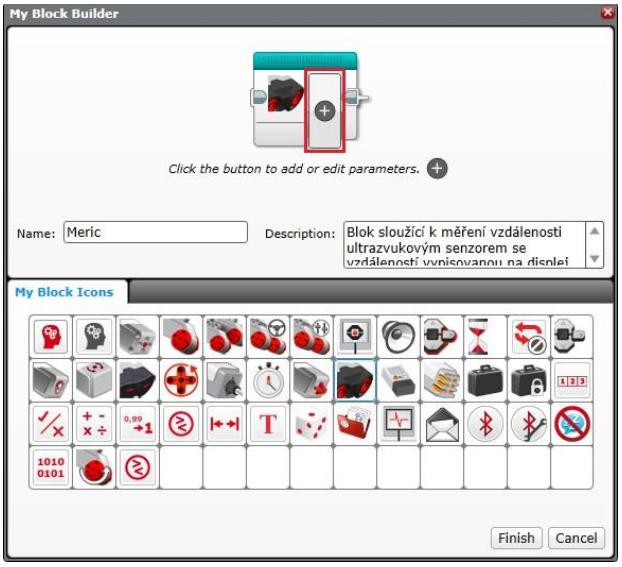
Umístění volby pro vytváření vlastních metod

Funkci pro vytvoření vlastních metod nalezneme v horním menu v záložce Tools (Nástroje) pod volbou My Block Builder (Tvůrce mých vlastních bloků). Dříve než začneme vlastní blok vytvářet, si musíme označit část zdrojového kódu, která bude v novém bloku obsažena.



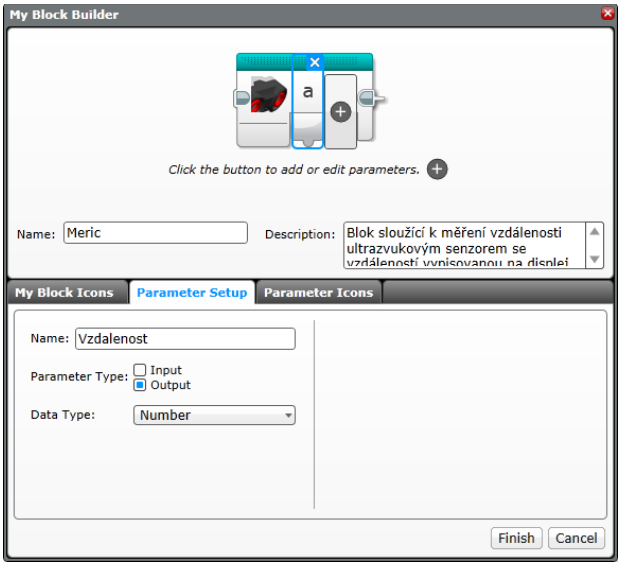
Úprava vzhledu bloku

V průvodci je nejprve nutné zadat unikátní název bloku a jeho popis, který nám v budoucnu usnadní orientaci v tom, co nový blok vykonává. Zvolit si můžeme také ikonu bloku z připravené palety obrázků. Následně můžeme přejít k nastavení vstupních a výstupních portů (pokud je chceme u bloku využít). Přidáme je pomocí tlačítka +, umístěného na náhledu bloku.

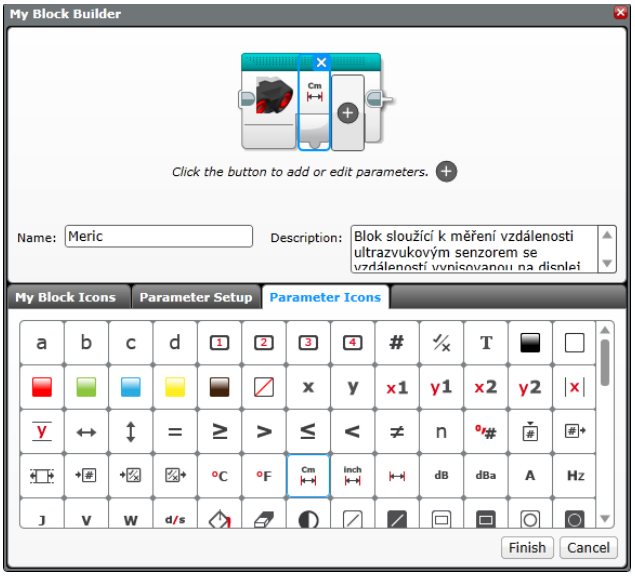


Nastavení vstupních a výstupních portů bloku

Po přidání libovolného počtu portů můžeme přejít k jejich nastavení. Zvolíme si název portu, jeho typ (vstupní nebo výstupní) a datový typ, kterého bude nabývat.

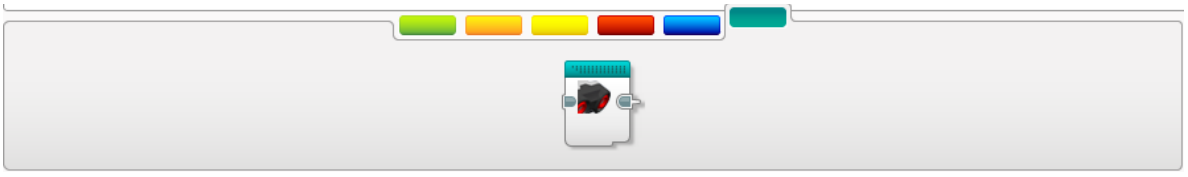


Na druhé záložce následně volíme z galerie ikonku, která se bude na portu zobrazovat.



Umístění nového bloku

Po kliknutí na tlačítko Finish se nový blok vytvoří a umístí se do záložky My Blocks (Mé bloky).



12 Závěrečné tipy

Doporučený multimediální materiál

Uživatelský manuál pro EV3 – odkaz viz. on-line kurz

Programovací prostředí RobotC

1 Základní informace o projektu

Název

Programovací prostředí RobotC

Anotace programu/zaměření/hlavní cíl

Programovací prostředí RobotC je programovací prostředí založené na syntaxi programovacího jazyka C. Je určeno pro vytváření programů pro ovládání širokého spektra robotických stavebnic. Jedná se například o VEX, RCX nebo Arduino. V této kapitole se seznámíte se samotným programovacím prostředím, jeho funkcemi a ovládáním a základní funkci si procvičíte díky množství připravených aktivit.

Cílová skupina

1. až 4. ročník SŠ a odpovídající ročníky gymnázií

Organizační podmínky

Spolupráce studentů ve dvoučlenných, maximálně tříčlenných skupinách.

Pomůcky

Robotická stavebnice EV3, programovací prostředí RobotC.

Časová náročnost (popř. jak je možné program rozložit – jedná- li se o celoroční program)

Odhadovaná doba - zhruba 15 - 20 vyučovacích hodin.

Vazba na RVP

Rámcový vzdělávací program pro gymnázia.

Mezipředmětové vazby

Informatika (informační a komunikační technologie), elektrotechnika.

2 Motivační rámec projektu

Text:

Grafická programovací prostředí typu EV3 umožňují nezkušeným uživatelům nahlédnout do tajů programování s pomocí robotické stavebnice. Logickým uspořádáváním a propojováním programových bloků zjednodušují první kroky a uvedení do problematiky pro nezkušené programátory.

Programovací prostředí RobotC od společnosti Robomatter, jehož syntaxe je založena na principu programovacího prostředí C, můžeme považovat za další krok na cestě k pochopení složitějších principů programování. Díky širším možnostem, které oproti grafickým programovacím prostředím nabízí, je možné vytvářet složitější programové konstrukty. Podobnost se syntaxí programovacího jazyka C může být také jedním z prvních kroků k pochopení tohoto jazyka. Prostředí RobotC není vyvíjeno pouze pro platformu LEGO Mindstorms EV3, ale i jiné (např. VEX, RCX nebo Arduino).

Doporučený multimediální materiál

Video viz. on-line kurz

Oficiální stránky výrobce programovacího prostředí: vice zde (odkaz viz. on-line kurz)

On-line dostupná uživatelská příručka pro RobotC: vice zde (odkaz viz. on-line kurz)

3 Poznámky k využití přístrojů

Pro tvorbu úvodu do práce s programovacím prostředím RobotC a sestavení úloh byla použita základní sada stavebnice EV3. Další informace o ní nalezenete na následujících odkazech:

Informace na oficiálních internetových stránkách výrobce (odkaz viz. on-line kurz)

On-line uživatelský manuál pro programovací prostředí RobotC: (odkaz viz. on-line kurz)

4 Projektový deník

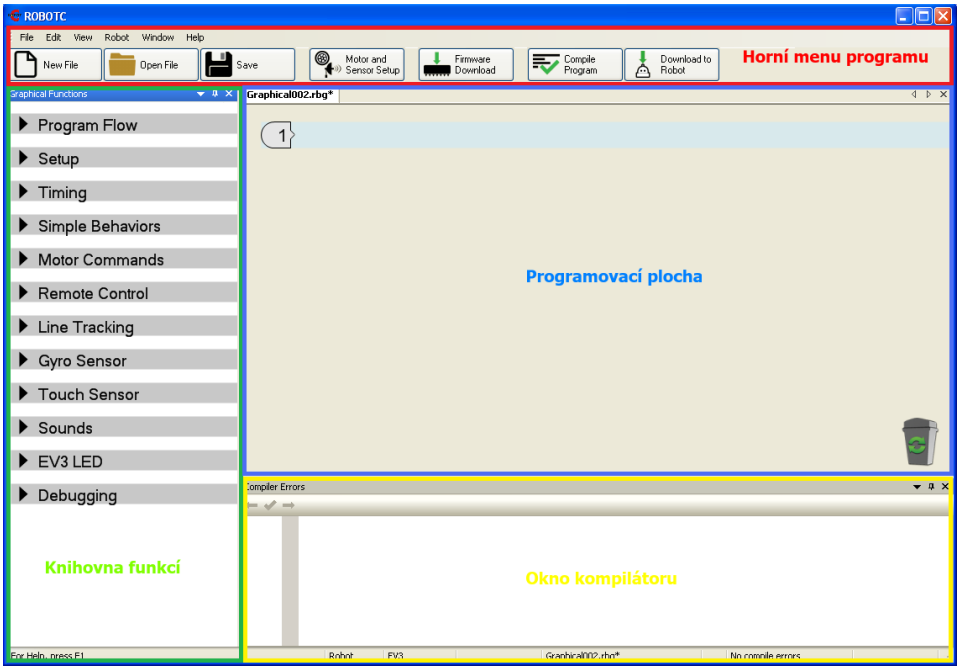
Projektový deník slouží k evidenci postupu žáků ve studiu kapitoly. Tabulky jsou uzpůsobeny tak, aby si žák měl možnost poznamenat problémy, které musel při řešení programové úlohy řešit, a co mu činilo u příkladu největší obtíže. Po kontrole zápisu a příslušného programu, případně modelu, vyučujícím je do tabulky zapsáno hodnocení splnil, nebo nesplnil. V případě potřeby může být doplněna také známka.

Projektový deník ke stažení (ve formátu PDF) v on-line kurzu. Také je přílohou této tiskové opory.

5 Grafické programovací prostředí RobotC

Aktuální verze programovacího prostředí RobotC (4.X) neumožňuje pouze vytvářet programový kód zápisem programovacího jazyka založeného na jazyce C, jako tomu bylo dříve, aktuálně obsahuje také grafický režim zápisu programového kódu.

Popis prostředí



Na ilustračním obrázku můžete vidět náhled grafické verze programovacího prostředí RobotC. Můžeme si jej rozdělit do čtyř základních částí:

Horní menu programu

Horní část programovacího prostředí (označena červeně) obsahuje standardní menu pro práci se souborem, editace a různé nastavení prostředí. Ve spodní části menu je umístěno několik pohotovostních tlačítek pro usnadnění vytváření programu. Jedná se o často využívané funkce při tvorbě programu.

New file - vytvoření nového programu.

Open file - otevření existujícího souboru.

Save - uložení programu.

Motor and Sensor Setup - deklarace motorů a senzorů.

Firmware Download - nahrání firmware.

Compile Program - kompilace programu.

Download to Robot - nahrání programu do řídicí jednotky EV3.

Knihovna funkcí

V levé části prostředí (označeno zeleně) se nachází knihovna programových funkcí, které je možné drag and drop způsobem přetáhnout na programovací plochu.

Programovací plocha

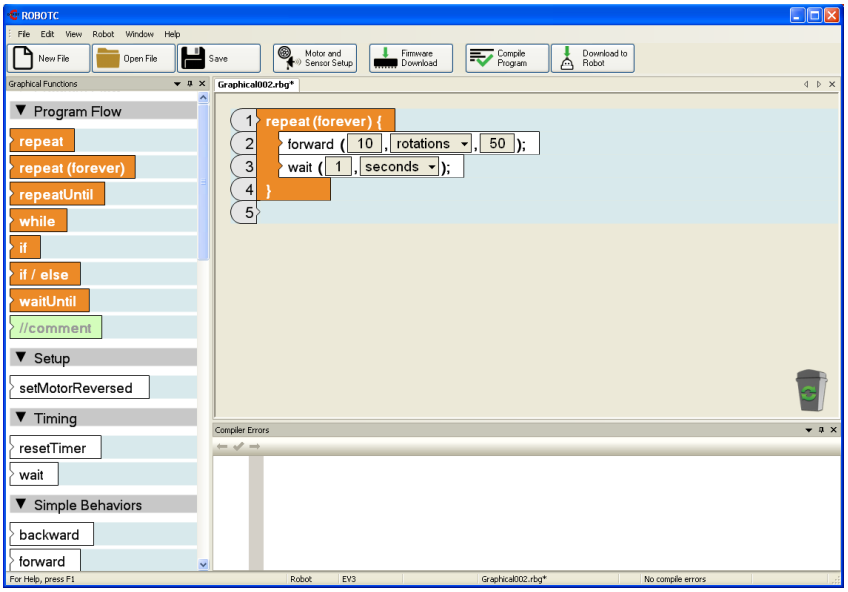
Největší část okna programovacího prostředí určená pro tvorbu programu pomocí skládání a vnořování jednotlivých grafických příkazů a funkcí.

Okno kompilátoru

Okno, ve kterém se zobrazují chyby a upozornění po kompilaci programu.

6 Základy tvorby programu

Grafická verze programovacího prostředí RobotC obsahuje v levé části knihovnu příkazů a funkcí, které je možné přetáhnout a umístit na programovací plochu. Na ilustračním obrázku můžete vidět příklad zápisu jednoduchého nekonečného cyklu. Jednotlivé funkce obsahují rozbalovací nabídky, ze kterých je možné volit parametry nebo měnit vstupní hodnoty.



Nevýhody grafické verze programovacího prostředí RobotC

Grafická verze programovacího prostředí RobotC neobsahuje všechny funkce, které obsahuje verze s textovým zápisem programového kódu. Zcela například chybí možnost jakéhokoliv výpisu na displej. Dále je také citelná absence vytváření proměnných a konstant.

Využitelnost prostředí

Programovací prostředí se dá využít pro základní výuku algoritmizace. Žáci se s jeho pomocí mohou naučit vytvářet jednoduché postupy pro řešení jednoduchých problémů. V následujících kapitolách naleznete náměty na aktivity pro grafickou verzi programovacího prostředí. Jedná se o vybrané aktivity, se kterými se setkáte v části věnované programovacímu prostředí EV3. Můžete tak porovnat jejich zápis v jiném programovacím prostředí.

6.1 Aktivita 1 - Ruční mixér

Zadání

Téma	Ruční mixér s použitím dotykového senzoru	
Tematický celek	Programovací prostředí RobotC (grafická verze prostředí)	
Motivační rámec	Dotykový senzor se hodí pro několik možností využití. Typicky se využívá jako tlačítko. Jeho velmi užitečnou vlastností je možnost rozeznávat tři stavy stisku. Tato aktivita se věnuje jeho použití jako spouštěče určité činnosti jiného zařízení.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Úkolem této aktivity je seznámit studenty s možnostmi použití dotykového senzoru. Ten je v úkolu použit jako tlačítko ručního mixéru. Mixér je poháněn pomocí středního motoru. Při ovládání rozlišuje tři stavy. Po prvním stisku a uvolnění se roztočí pomaleji, po dalším stisku maximální rychlostí a při třetím stisku se vypne.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Studenti se naučí využívat dotykový senzor a rozeznávat jeho možné stavy.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Základní ovládání programovacího prostředí RobotC, základy algoritmizace.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Stavba modelu.	Spolupráce studentů ve skupinách, koordinace činnosti
15 minut	Tvorba programu a průběžné	Spolupráce studentů ve skupinách, koordinace činnosti
5 minut	Závěrečné testování funkčnosti.	Spolupráce studentů ve skupinách, koordinace činnosti
Hodnocení	Hodnocena bude průběžná práce na úloze, spolupráce studentů ve skupině a kvalita výsledného modelu a programu.	

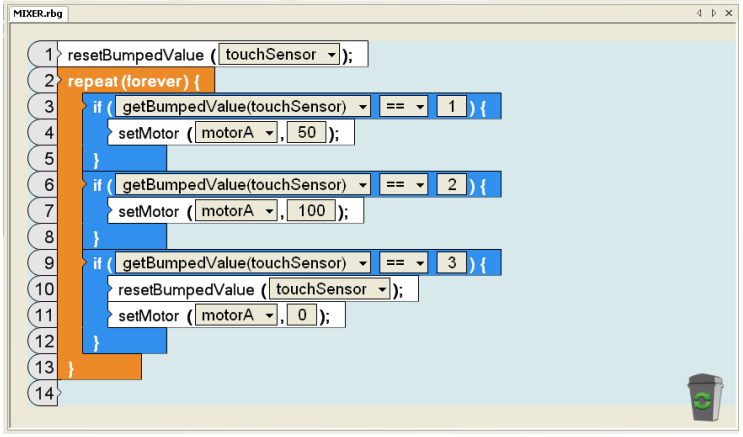
Aktivita si klade za cíl vás seznámit s funkcemi dotykového senzoru. V následující úloze se naučíte využít dotykový senzor jako spouštěč či přepínač určité reakce. Váš úkol je následující:

1. Postavte vhodný model ručního mixéru, který bude ovládán pomocí tlačítka dotykového senzoru.
2. Po prvním stisknutí se mixér roztočí pomaleji, při druhém zrychlí na maximum a při třetím zastaví.

Řešení pro grafickou verzi programovacího prostředí RobotC

Řešení v grafické verzi RobotC se liší od textového zápisu. Nesmíte zapomenout na správnou deklaraci motorů a senzoru tak, aby byla zajištěna funkčnost programu.

Neustálé provádění programu zaručíme cyklem repeat (forever). Následně v programu ověřujeme počet stisknutí tlačítek. Na základě zjištěné hodnoty je ovlivňována rychlost otáčení motoru, který pohání ruční mixér. V úloze si procvičíte práci s cykly, podmínkami, vyhodnocování hodnoty zjištěné senzorem a také nastavení rychlosti otáčení motoru.



Vysvětlivky ke zdrojovému kódu

resetBumpedValue () - nulování čítače stisků tlačítka dotykového senzoru

repeat (forever) - cyklus s nekonečným počtem vykonávání

if () - podmínka

getBumpedValue() - detekce počtu stisknutí tlačítka dotykového senzoru

setMotor () - ovládání rychlosti otáčení motoru

Výsledný program ke stažení ve formátu .RBG

Program ke stažení viz. on-line kurz

6.2 Aktivita 2 - Výstražný semafor se zvukovou signalizací

Zadání

Téma	Výstražný semafor se zvukovou signalizací	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Signalizace železničního přejezdu upozorňuje řidiče, že se k přejezdu blíží vlak, a je tudíž životu nebezpečné do něj vjíždět. Podoba železniční signalizace je různá. V této úloze si vytvoříte program, který bude simulovat funkci světelné signalizace na železničním či jiném přejezdu. Přidanou hodnotou bude zvuková signalizace upozrňující na blížící se nebezpečí.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím EV3.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je využít zvukovou a světelnou signalizaci řídicí jednotky jako signalizátor blížícího se nebezpečí na železničním přejezdu. Detektorem blížícího se vlaku se stane ultrazvukový senzor, který pomocí naměřené vzdálenosti spustí světelnou a zvukovou signalizaci.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat světelnou a zvukovou signalizaci řídicí jednotky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce se světelnou a zvukovou signalizací.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
40 minut	Tvorba programu a jeho průběžné	Spolupráce studentů ve skupinách, koordinace činnosti
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

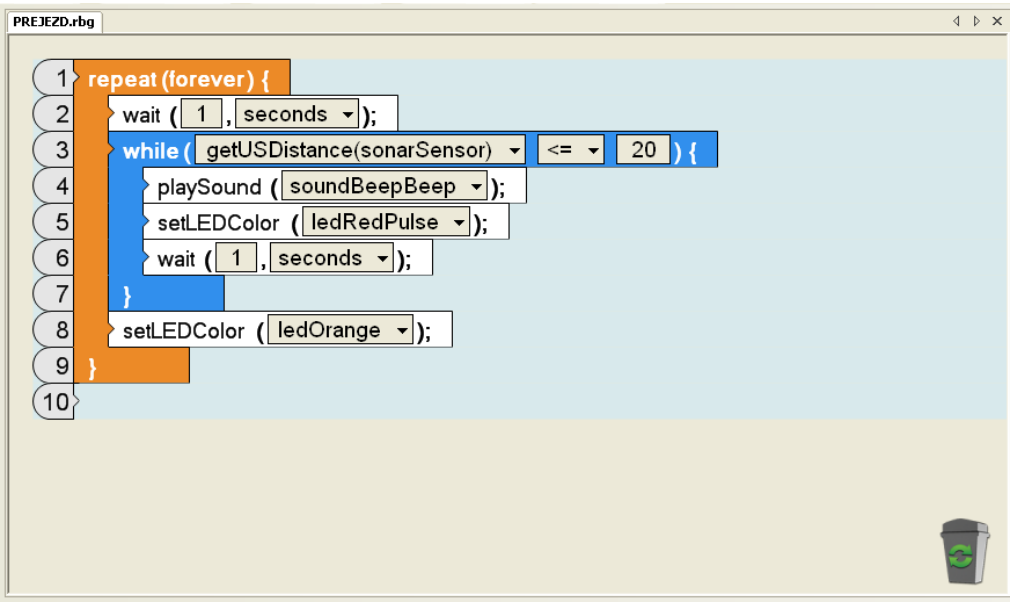
Zadání

Využijte řídicí jednotku k vytvoření programu, který bude simulovat signalizaci blížícího se vlaku na železničním přejezdu. Požadavky na tvorbu budou následující:

1. Použijte ultrazvukový senzor pro detekci blížícího se vlaku.
2. Při naměření kritické vzdálenosti se spustí přerušovaná světelná a také zvuková signalizace.
3. Jakmile se pomyslný vlak vzdálí opět za kritickou vzdálenost, signalizace se vypne.

Řešení pro grafickou verzi programovacího prostředí RobotC

Na obrázku můžete vidět zápis programového kódu úlohy v grafickém zápisu programovacího prostředí RobotC. Celý program se vykonává neustále dokola, proto je použit cyklus repeat(forever). Dále je v úloze použita výstražná zvuková signalizace a také podsvícení tlačítek řídicí jednotky. Chování výstrahy je řízeno pomocí vzdálenosti snímané ultrazvukovým senzorem.



Vysvětlivky ke zdrojovému kódu

repeat (forever) - cyklus s nekonečným počtem vykonávání

while () - cyklus s podmínkou na začátku

wait () - oddálení vykonávání příkazu

getUSDistance () - funkce pro získání hodnoty snímané ultrazvukovým senzorem

playSound () - příkaz pro přehrání zvuku

setLEDColor () - nastavení barvy, kterou bude svítit čelní panel řídicí jednotky

Výsledný program ke stažení ve formátu .RBG

Program ke stažení viz. on-line kurz

6.3 Aktivita 3 - Detektor pádu pevného disku

Téma	Detektor pádu disku	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Nenadálé problémy se spuštěním počítače mohou někdy pramenit z poruchy pevného disku počítače. Stát se tak může i v jiných případech. Například při pádu spuštěného notebooku na zem. Ať již je příčina jakákoliv, ve velké většině případů nastane nepříjemná ztráta dat. Pevné disky ovšem obsahují zařízení, které při detekci pádu dokáže bezpečně zaparkovat zápisovou hlavu tak, aby nedošlo k poškození disku. My si takové zařízení zkusíme vytvořit díky robotické stavebnici za pomoci gyroskopického senzoru.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve sestaví jednoduchý model pevného disku. Uvnitř se pokusí vytvořit pohyblivý mechanismus, který bude simulovat jeho funkci (např. motor nebo pohyblivé rameno). Zařízení pro detekci pádu bude představovat gyroskopický senzor. Následně vytvoří program, který při prudké změně polohy zastaví chod disku.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat gyroskopický senzor.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s barevným senzorem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu disku.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

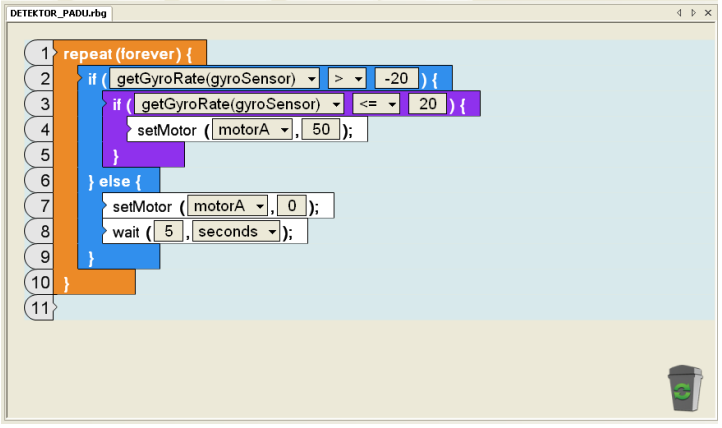
Zadání

Vytvořte funkční model pevného disku se zařízením pro detekci pádu. Požadavky na konstrukci modelu a program budou následující:

1. Sestavte jednoduchý model pevného disku.
2. Uvnitř bude umístěn pohyblivý mechanismus poháněný například motorem.
3. Pro detekci pádu použijte gyroskopický senzor.
4. Jakmile bude detekována prudká změna polohy, chod disku se zastaví.

Řešení pro grafickou verzi programovacího prostředí RobotC

Obrázek ilustruje zápis grafického programového kódu programu pro ovládání detektoru pádu disku.
Pro nekonečné provádění programu je použit nekonečný cyklus repeat(forever). Následně je detekována poloha disku. Při pouze drobném chvění gyroskopického senzoru pokračuje disk v chodu. Pokud je detekován prudký otřes nebo volný pád, je motor na 5 vteřin vypnut.



Vysvětlivky ke zdrojovému kódu

repeat (forever) - cyklus s nekonečným počtem vykonávání
if () - podmínka

getGyroRate () - zjištění hodnoty snímané senzorem

setMotor () - nastavení rychlosti otáčení motoru

wait () - oddálení vykonávání příkazu

Výsledný program ke stažení ve formátu .RBG

Program ke stažení viz. on-line kurz

7 Textově orientované programovací prostředí

Charakteristika prostředí:

- program vytvářen textovým zápisem programovacího jazyka,
- syntaxe jazyka založena na bázi programovacího jazyka C,
- tvůrce prostředí - Robomatter, Inc.

video viz. on-line kurz

Popis jednotlivých částí RobotC

Horní menu programu:

- nalezneme jej v horní části programovacího prostředí,
- obsahuje základní volby pro práci se souborem (vytvoření nového programu, otevření nebo uložení),
- zobrazení různých plovoucích panelů nástrojů,
- nastavení využívané platformy, aktualizace firmware, konfigurace motorů a senzorů.

Akční ikony:

New File - vytvoření nového souboru,
Open File - otevření existujícího programu, *Save* - uložení vytvářeného programu,
Fix Formatting - naformátuje ve zvoleném souboru veškeré odsazení a závorky,
Motor and Sensor Setup - volba sloužící k nastavení motorů a senzorů (přiřazení na porty, pojmenování, volba režimu), *Firmware Download* - nahrání firmware do řídíïcí jednotky,
Compile Program - kompilace programu (bez nahrání programu do řídicí jednotky), *Download to Robot* - kompilace programu s následným nahráním do paměti řídicí jednotky,
Start - spustí provádění programu (zobrazí se až po nahrání programu do paměti řídicí jednotky),
Stop - zastaví provádění spuštěného programu (zobrazí se až po nahrání programu do paměti řídicí jednotky).

Knihovna funkcí:

- umístěna vlevo,
- obsahuje stromovou strukturu všech kategorií, kde jsou umístěny využitelné funkce a příkazy.

Programovací plocha:

- prostor pro zápis programu,
- pro snazší orientaci v programu je vlevo umístěno číslování jednotlivých řádků programu.

Okno kompilátoru:

- na tomto místě se zobrazují chyby v programu zjištěné při kompilaci.

Poznámka

V následujících podkapitolách naleznete návrhy aktivit na procvičení práce s moduly stavebnice LEGO Mindstorms EV3 v programovacím prostředí RobotC. Úlohy jsou ve většině totožné s úlohami obsaženými v kapitole o programovacím prostředí EV3. Díky tomu si můžete procvičit tvorbu paralelně jak v grafickém, tak i textovém programovacím prostředí.

7.1 Aktivita 1 - Lopatková turbína

Téma	Lopatková turbína	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Trendem dnešní doby je využívání obnovitelných zdrojů energie. Zásoba přírodních zdrojů není nevyčerpatelná, a tak musíme stále hledat vhodné alternativy. Během slunečných dnů můžeme využívat například solární panely. Jak ovšem vyrábět elektřinu ve stejném objemu i v období, kdy není světla dostatek nebo v noci? V této úloze si sestrojíte jednoduchou turbínu fungující jako záložní způsob výroby energie v situacích, kdy je světla nedostatek.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve vytvoří jednoduchý model lopatkové turbíny. Ta bude poháněna pomocí střední motoru. Její chod bude řízen barevným senzorem, který pomocí detekce okolního světla bude řídit rychlost otáčení.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat střední motoru k pohonu rotačních součástí a řídit rychlost otáčení v závislosti na jiném zařízení.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s barevným senzorem.	
Mezipředmětové vztahy	Fyzika (světlo), Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Vytvořte funkční model turbíny jako záložního zdroje výroby elektrické energie. Požadavky na konstrukci modelu a program budou následující:

1. Lopatky turbíny budou poháněny středním motorem.
2. Řízení rychlosti otáčení bude prováděno díky barevnému senzoru.
    a. Čím nižší bude intenzita okolního světla, tím se bude turbína otáčet rychleji.
3. V programu vhodně nastavte prahovou hodnotu dostatečného slunečního svitu, při kterém se turbína zastaví.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.2 Aktivita 2 - Kuchyňská minutka

Téma	Kuchyňská minutka	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Pravděpodobně každý z nás se někdy v kuchyni setkal s kuchyňskou minutkou. Malým zařízením sloužícím k měření času při přípravě pokrmů. Využívaly ji naše babičky i maminky. Doba pokročila a z klasických manuálních se vyvinuly i pokročilejší digitální, které jsou často opatřené displejem. Vše potřebné k tomu, abychom podobnou minutku vytvořili z robotické stavebnice, ovšem máme k dispozici i my. Postačí nám k tomu displej řídicí jednotky a otočný mechanismus zajistí servomotor.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve vytvoří vhodnou konstrukci kuchyňské minutky sestávající ze servomotoru opatřeného dobře ovladatelným otáčecím mechanismem pro zajištění pohodlného nastavení času měření. Odpočítávání začne ve chvíli, kdy bude stisknuto tlačítko připevněného dotykového senzoru.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat servomotor k ovládání pohyblivých součástí a pracovat s měřením a ovlivňováním vykonaného počtu otáček.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s dotykovým senzorem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
15 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
30 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
10 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Vytvořte funkční model kuchyňské minutky. Požadavky na konstrukci modelu a program budou následující:

1. Otočný mechanismus minutky bude realizován pomocí servomotoru.
2. Natočením mechanismu se díky detekci počtu otáček vyhodnotí, jak dlouhý časový úsek se má měřit.
3. Nastavovovaný čas by se měl zobrazit na displeji.
4. Při měření času by se měla minutka otáčet a konec odpočítávání by měl oznámit zvukový signál.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.3 Aktivita 3 - Řízení pásového transportéru

Téma	Řízení pásového transportéru	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	V těžko dostupných terénech na nestabilním podkladu často kolová vozidla nezvládnou bez problému průchod terénem. Je potřeba proto využít vozidla pásová. Využívají se v armádě, ale i mezi civilními občany. Typickým příkladem je sněžná rolba nebo pásový transportér. V této aktivitě si podobné vozidlo sestavíme a hlavně se je naučíme ovládat.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve vytvoří model pojízdného robota opatřeného pásy. Robot bude opatřen dvěma dotykovými senzory. V případě, že budou stisknuta tlačítka obou senzorů současně, robot se bude pohybovat vpřed. Při stisku pouze jediného z tlačítek se bude robot pohybovat na zvolenou stranu.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat alternativní způsob pohonu servomotorů (pomocí pásů).	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost možností řízení motorů, práce s dotykovým senzorem a znalost práce s cykly a podmínkami.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
20 minut	Tvorba modelu robota.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocen bude sestavený model robota a úplnost a funkčnost vytvářeného programu.	

Zadání

Vytvořte funkční model robota poháněného pásy. Požadavky na konstrukci modelu a program budou následující: robot musí být dobře

manévrovatelný a pásy na modelu dobře upevněny,
natáčení robota do stran bude prováděno dvěma dotykovými senzory plnícími funkci joysticku, volte vhodnou rychlost otáčení motorů pro dobrou pohyblivost modelu.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.4 Aktivita 4 - Železniční přejezd

Téma	Železniční přejezd	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Signalizace železničního přejezdu upozorňuje řidiče, že se k přejezdu blíží vlak, a je tudíž životu nebezpečné do něj vjíždět. Podoba železniční signalizace je různá. V této úloze si vytvoříte program, který bude simulovat funkci světelné signalizace na železničním přejezdu. Přidanou hodnotou bude zvuková signalizace upozorňující na blížící se nebezpečí.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC (textová	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je využít zvukovou a světelnou signalizaci řídicí jednotky jako signalizátor blížícího se nebezpečí na železničním přejezdu. Detektorem blížícího se vlaku se stane ultrazvukový senzor, který pomocí naměřené vzdálenosti spustí světelnou a zvukovou signalizaci.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat světelnou a zvukovou signalizaci řídicí jednotky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce se světelnou a zvukovou signalizací.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
40 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Využijte řídicí jednotku k vytvoření programu, který bude simulovat signalizaci blížícího se vlaku na železničním přejezdu.
Požadavky na tvorbu budou následující:

1. Použijte ultrazvukový senzor pro detekci blížícího se vlaku.
2. Při naměření kritické vzdálenosti se spustí přerušovaná světelná a také zvuková signalizace.
3. Jakmile se pomyslný vlak vzdálí opět za kritickou vzdálenost, signalizace se vypne.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.5 Aktivita 5 - Kreslicí tabulka

Téma	Kreslící tabulka	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Mnoho z nás si z dětství pamatuje ruční kreslicí tabulku s displejem a dvěma ovládajícími kolečky pro kreslení. Jedno kolečko pro kreslení v horizontálním a druhé ve vertikálním směru. Jelikož tyto hračky nahradily jiné, mnohem propracovanější, setkáme se s takovou tabulkou jen zřídka. Není pro nás ovšem žádný problém si takovou tabulku pomocí robotické stavebnice vyrobit přímo pomocí řídicí jednotky. Jako ovládací prvky nám postačí její tlačítka.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je vytvořit kreslicí tabulku pomocí displeje řídicí jednotky. Při stisku směrových tlačítek se bude vždy vykreslovat do požadovaného směru. Středové tlačítko slouží k mazání plochy displeje.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí vykreslovat či vypisovat různé prvky na plochu displeje řídicí jednotky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s tlačítky řídicí jednotky.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
40 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňujích multimediálních prvků.	

Zadání

Vytvořte funkční model kreslicí tabulky. Požadavky na tvorbu budou následující:

1. Pro kreslení využijte plochu displeje řídicí jednotky.
2. Ovládání při vykreslování budou obstarávat tlačítka řídicí jednotky.
3. Prostřední tlačítko bude sloužit k mazání plochy displeje.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.6 Aktivita 6 - Ruční mixér

Téma	Ruční mixér s použitím dotykového senzoru	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Dotykový senzor se hodí pro několik možností využití. Typicky se využívá jako tlačítko. Jeho velmi užitečnou vlastností je možnost rozeznávat tři stavy stisku. Tato aktivita se věnuje jeho použití jako spouštěče určité činnosti jiného zařízení.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Úkolem této aktivity je seznámit studenty s možnostmi použití dotykového senzoru. Ten je v úkolu použit jako tlačítko ručního mixéru. Mixér je poháněn pomocí středního motoru. Při ovládání rozlišuje tři stavy. Po prvním stisku a uvolnění se roztočí pomaleji, po dalším stisku maximální rychlostí a při třetím stisku se vypne.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Studenti se naučí využívat dotykový senzor a rozeznávat jeho možné stavy.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s dotykovým senzorem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Stavba modelu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Tvorba programu a průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Závěrečné testování funkčnosti.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na úloze, spolupráce studentů ve skupině a kvalita výsledného modelu a programu.	

Zadání

Aktivita si klade za cíl vás seznámit s funkcemi dotykového senzoru. V následující úloze se naučíte využít dotykový senzor jako spouštěč či přepínač určité reakce. Váš úkol je následující:

1. Postavte vhodný model ručního mixéru, který bude ovládaný pomocí tlačítka dotykového senzoru.
2. Po prvním stisknutí se mixér roztočí pomaleji, při druhém zrychlí na maximum a při třetím zastaví.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.7 Aktivita 7 - Rozpoznávač barev

Téma	Rozpoznávač barev s využitím barevného senzoru	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Ne každý z nás má barevné cítění a okamžitě rozpozná, o jakou barvu se jedná. Pociťujeme to hlavně při nákupu oblečení. Barev existuje tolik, že se v nich laik téměř nevyzná. Díky robotické stavebnici si ale dokážeme sestrojit jednoduchý rozpoznávač barev, který nám alespoň základní barvy dokáže určit.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC (textová verze).	
Stručný popis aktivity s využitím přístroje	Úkolem studentů je vytvořit jednoduchý model ručního rozpoznávače barev, který bude opatřený barevným senzorem. Ten bude zjišťovat barevný odstín snímaného materiálu a barvu následně vypíše na displej řídicí jednotky.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Studenti se naučí pracovat s barevným senzorem, seznámí se s jeho možnými režimy a s jeho využitím sestrojí jednoduché zařízení k rozpoznávání barev.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s barevným senzorem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace), fyzika (světlo).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Stavba modelu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Tvorba programu a průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Závěrečné testování a praktické ověření funkčnosti.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a kvalita a funkčnost sestaveného modelu a vytvořeného programu.	

Zadání

V této aktivitě je vaším úkolem sestavit jednoduchý ruční rozpoznávač barev. Pro úspěšné splnění úlohy naplňte následující požadavky:

1. Sestavte model ručního rozpoznávače barev, který bude možné držet v jedné ruce (využijte barevný senzor).
2. Jakmile namíříte senzor na povrch, jehož barvu potřebujeme zjistit, informace o zjištěné barvě se vypíše na displej řídicí jednotky.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.8 Aktivita 8 - Detektor pádu pevného disku

Téma	Detektor pádu disku	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Nenadálé problémy se spuštěním počítače mohou někdy pramenit z poruchy pevného disku počítače. Stát se tak může i v jiných případech. Například při pádu spuštěného notebooku na zem. Ať již je příčina jakákoliv, ve velké většině případů nastane nepříjemná ztráta dat. Pevné disky ovšem obsahují zařízení, které při detekci pádu dokáže bezpečně zaparkovat zápisovou hlavu tak, aby nedošlo k poškození disku. My si takové zařízení zkusíme vytvořit díky robotické stavebnici za pomoci gyroskopického senzoru.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Žáci si nejprve sestaví jednoduchý model pevného disku. Uvnitř se pokusí vytvořit pohyblivý mechanismus, který bude simulovat jeho funkci (např. motor nebo pohyblivé rameno). Zařízení pro detekci pádu bude představovat gyroskopický senzor. Následně vytvoří program, který při prudké změně polohy zastaví chod disku.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat gyroskopický senzor.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost funkce a režimů gyroskopického senzoru.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace úloh).	
Časový plán	Fáze činnosti s	Metody a formy, motivace
15 minut	Tvorba modelu disku.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
20 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Testování funkčnosti modelu a programu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude úplnost a přehlednost vyplněných informací a kvalita doplňkových multimediálních prvků.	

Zadání

Vytvořte funkční model pevného disku se zařízením pro detekci pádu. Požadavky na konstrukci modelu a program budou následující:

1. Sestavte jednoduchý model pevného disku.
2. Uvnitř bude místěn pohyblivý mechanismus poháněný například motorem.
3. Pro detekci pádu použijte gyroskopický senzor.
4. Jakmile bude detekována prudká změna polohy, chod disku se zastaví.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.9 Aktivita 9 - Turniket

Téma	Inteligentní turniket	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Na hudebních koncertech a různých sportovních akcích se zjišťuje různými způsoby aktuální počet návštěvníků. Většinou se tomu tak děje díky tuniketům, které sčítají počet lidí, kteří jimi projdou. V následující úloze si takový turniket vytvoříme za pomoci ultrazvukového senzoru.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC (textová	
Stručný popis aktivity s využitím přístroje	V této úloze studenti vytvoří model jednoduchého turniketu, který sčítá počet návštěvníků kulturní akce. Ultrazvukový senzor bude přesně snímat prostor určení pro příchod do areálu. Jakmile jeho vysílaný signál protne některý z příchozích návštěvníků, bude započítán. Počet návštěvníků se bude postupně přičítat a vypisovat na displej.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Studenti se naučí pracovat s ultrazvukovým senzorem a naučí se zpracovávat jím zjištěná data.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost funkce a režimů ultrazvukového senzoru.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
10 minut	Tvorba modelu turniketu.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
15 minut	Tvorba programu a průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
5 minut	Ověření funkčnosti a závěrečné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a kvalita sestaveného modelu a funkčnost vytvořeného programu.	

Zadání

V této aktivitě se seznámíte s funkcí ultrazvukového senzoru. Vaším úkolem je vytvořit inteligentní turniket pro kulturní akce, který sčítá počet návštěvníků. Požadavky na jeho funkčnost jsou následující:

1. Vytvořený model musí být jednoduchý a plně funkční (využijte ultrazvukový senzor).
2. Průchod turniketem musí být snímám ultrazvukovým senzorem, který bude detekovat každého příchozího návštěvníka.
3. Jakmile návštěvník turniketem projde, bude zaznamenán.
4. Aktuální počet návštěvníků se bude vypisovat na displeji řídicí jednotky.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.10 Aktivita 10 - Bomba

Téma	Bomba	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Práce pyrotechnika není záviděníhodná. Stojí za ní roky zkušeností, znalostí elektrotechniky a hlavně pevné nervy. Zkušený pyrotechnik ovšem dokáže vyhodnotit povahu výbušniny a její funkci a zneškodnit ji. Vaším úkolem v této aktivitě ale bude vytvořit nepředvídatelnou bombu, jejíž chování nebude možné zjistit. Bude záviset pouze na štěstí, zda se ji povede zneškodnit, či nikoliv.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je vytvořit program, který bude simulovat bombu. Potenciálnímu pyrotechnikovi se na displeji zobrazí výzva, aby bombu zneškodnil jedním ze tří nabízených tlačítek. U žádného z nich ovšem nebude zaručeno, že je správné. Význam tlačítek se totiž bude náhodně generovat. Program žáci doplní o vhodnou zvukovou signalizaci a výpis na displej.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Studenti se naučí pracovat s tlačítky řídicí jednotky a naučí se vyhodnocovat jejich stisk a reagovat na něj.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s tlačítky řídicí jednotky.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a funkčnost vytvořeného programu.	

Zadání

V této aktivitě se naučíte využívat tlačítka řídicí jednotky. Vaším úkolem je vytvořit z řídicí jednotky pomyslný model bomby. Co pro úspěšnou realizaci musíte zvládnout?

1. Zvolte si alespoň tři tlačítka řídicí jednotky, která budou představovat dráty, které musí pyrotechnik přestříhnout.
2. Programově zajistěte, aby žádné tlačítko nebylo určeno ke zneškodnění bomby, ale aby se jejich funkce náhodně měnila.
3. Program vhodně graficky a zvukově ošetřete.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.11 Aktivita 11 - Detektor světla

Téma	Detektor světla	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Zařízení detekující úroveň světla v okolí nalezneme v mnoha mechanismech. Jedním z nich může být například automatické osvětlení, které na základě detekce světelných podmínek spouští pouliční osvětlení či nikoliv. My se v této aktivitě pokusíme takový detektor sestrojit. Využijeme k tomu barevný senzor, který režim detekce světla v okolí obsahuje.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je sestrojit a naprogramovat jednoduchý detektor světla, který rozlišuje světelné podmínky v okolí. Na displeji následně zobrazuje, zda je světla v okolí dostatek, či nikoliv.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Záci se naučí využívat cykly a podmínky.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s cykly a podmínky. Znalost režimů barevného senzoru.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a funkčnost vytvořeného programu.	

Zadání

V této aktivitě se naučíte využívat tlačítka řídicí jednotky. Vaším úkolem je vytvořit z řídicí jednotky pomyslný model bomby. Co pro úspěšnou realizaci musíte zvládnout?

1. Zvolte si alespoň tři tlačítka řídicí jednotky, která budou představovat dráty, které musí pyrotechnik přestříhnout.
2. Programově zajistěte, aby žádné tlačítko nebylo určeno ke zneškodnění bomby, ale aby se jejich funkce náhodně měnila.
3. Program vhodně graficky a zvukově ošetřete.

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

7.12 Aktivita 12 - Stopky

Téma	Stopky	
Tematický celek	Programovací prostředí RobotC	
Motivační rámec	Každý správný trenér potřebuje stopky. Využívají je trenéři vrcholových sportovců, závodníků, ale také například učitelé tělesné výchovy. Pomocí nich kontrolují výkony svých svěřenců a ověřují případné zlepšení v jejich sportovných výkonech. Jednoduché stopky si můžeme vytvořit také pomocí robotické stavebnice.	
Počet žáků	Skupina 8 - 10 studentů.	
Věk žáků	1. až 4. ročník SŠ a odpovídající ročníky gymnázií	
Pomůcky	Robotická stavebnice EV3 a počítače s nainstalovaným programovacím prostředím RobotC.	
Stručný popis aktivity s využitím přístroje	Úkolem žáků je sestrojit a naprogramovat jednoduchý model digitálních stopek. Ty budou měřit postupně tři kola, která závodník urazil, a na závěr vypíšou průměrný čas na kolo ze změřených tří kol. Jako měřicí tlačítko stopek může posloužit tlačítko dotykového senzoru nebo některé z tlačítek řídicí jednotky.	
Vhodné místo	Běžná učebna vybavená počítači s nainstalovaným programovacím prostředím RobotC.	
Cíle aktivity	Žáci se naučí využívat pole.	
Rozvíjené kompetence	Kompetence k učení, k řešení problémů.	
Předchozí znalosti	Znalost práce s polem.	
Mezipředmětové vztahy	Informační a komunikační technologie (algoritmizace).	
Časový plán	Fáze činnosti s přístrojem	Metody a formy, motivace
45 minut	Tvorba programu a jeho průběžné testování.	Spolupráce studentů ve skupinách, koordinace činnosti vyučujícím.
Hodnocení	Hodnocena bude průběžná práce na projektu a funkčnost vytvořeného programu.	

Zadání

Aktivita slouží k tomu, abyste se naučili využívat v programovacím prostředí RobotC pole. Za tímto účelem naprogramujete model stopek, které budou splňovat následující požadavky:

1. Zvolte si, čím budou stopky ovládány (např. tlačítko řídící jednotky nebo tlačítko dotykového senzoru).
2. Stopky se spustí po spuštění programu a po každém stisknutí tlačítka bude zaznamenán čas aktuálně měřeného kola.
3. Celkem budou zaznamenána tři kola a následně bude spočten průměrný čas na kolo (všechny časy budou vypsány na dipslej řídicí jednotky).

Výsledný program ke stažení ve formátu .C

Program ke stažení viz. on-line kurz

8 Tipy a triky

Vytváření vlastních metod

Při vytváření programu v RobotC nemusíme využívat pouze funkce obsažené v knihovně funkcí. Je možné si vytvářet také funkce vlastní, které lze následně použít na libovolných místech v programu. Díky tomu, že nebudeme funkci zapisovat opakovaně, zkrátíme a zpřehledníme svůj program. Zároveň si také usnadníme jeho vytváření.

Funkce bez návratové hodnoty

Prvním typem vlastní vytvářené funkce je taková funkce, která při svém zavolání nevrací žádnou hodnotu, pouze se provede určitá sekvence příkazů. Funkce se definuje klíčovým slovem ***void*** značícím funkci bez návratové hodnoty, za kterým je uveden její název, případně parametr uvedený v závorce. V těle funkce se poté uvádí kód, který se při jejím zavolání vykoná.

Na obrázku níže můžete vidět příklad funkce bez návratové hodnoty včetně jejího zavolání v hlavní části programu.

```
void vzdalenost (int x)
{
  if (SensorValue(ultrazvuk) > x)
  {
    motor[motorA] = 0;
    motor[motorB] = 0;
  }
}

task main()
{

  vzdalenost(20);

}
```

S návratovou hodnotu

Druhá možnost realizace vlastní funkce je taková, od které očekáváme získání určitého výsledku. Z toho důvodu je nutné v jejím zápisu definovat, jakého datového typu bude vrácený výsledek nabývat. Klíčovým slovem ***return*** na závěr určíme, jakou hodnotu chceme po zavolání funkce získat.

Na obrázku níže můžete vidět jednoduchý příklad deklarace funkce s návratovou hodnotou i s jejím následným zavoláním a použitím v hlavní části programu.

```
int vzdalenost (int x)
{
  x = (getUSDistance (ultrazvuk)) * 22;
  return x;
}


task main()
{

while (true)
{
  int x = 0;
  int vypocet = vzdalenost (x);
}

}
```

Dotaz task

Rozšiřující možnosti řízení umožňuje vlastní vytvořený dotaz task. Při jeho deklaraci v programu je definován klíčovým slovem task následovaným názvem dotazu a jeho parametry. Jeho zavolání v hlavní části programu je poté provedeno příkazem starTask, jehož parametrem v závorce je název dotazu. Od klasické funkce se dotaz liší možnostmi jeho řízení. Jedná se například o řízení výpočetního výkonu přiřazeného pro vykonávání funkce nebo prioritu jejího vykonávání.

Jednoduchý příklad deklarace a následného zavolání dotazu můžete vidět na následujícím obrázku.

```
task detekce ()
{
  while(true)
  {
    if(getGyroRate(gyro) > -20 & getGyroRate(gyro) < 20)
    {
      motor[motorA] = 20;
    }
    else
    {
      motor[motorA] = 0;
      wait1Msec(5000);
    }
  }
}


task main()
{

  startTask(detekce);

}
```

9 Závěrečné tipy

Doporučený multimediální materiál

Nabídka základních i doplňkových komponent na prodejním webu české společnosti EDUXE s.r.o. (odkaz viz. on-line kurz)

Oficiální stránky společnosti Robomatter - tvůrce programovacího prostředí RobotC (odkaz viz. on-line kurz)

Projektový deník - Programovací prostředí EV3

Jméno:
Třída:

Programovací prostředí EV3	*Stručný postup (problémy řešené při práci, způsob řešení)*	*Hodnocení vyučujího (splněno/nesplněno)*
Aktivita 1 - Příprava projektu Datum:		
Aktivita 2 - Lopatková turbína Datum:		
Aktivita 3 - Kuchyňská minutka Datum:		
Aktivita 4 - Řízení pásového transportéru Datum:		
Aktivita 5 - Železniční přejezd Datum:		
Aktivita 6 - Kreslící tabulka Datum:		

Programovací prostředí EV3	*Stručný postup (problémy řešené při práci, způsob řešení)*	*Hodnocení vyučujího (splněno/nesplněno)*
Aktivita 7 - Ruční mixér Datum:		
Aktivita 8 - Rozpoznávač barev Datum:		
Aktivita 9 - Detektor pádu pevného disku Datum:		
Aktivita 10 - Turniket Datum:		
Aktivita 11 - Bomba Datum:		
Aktivita 12 - Detektor světla Datum:		
Aktivita 13 - Stopky Datum:		

Projektový deník - Grafické programovací prostředí RobotC

Jméno:
Třída:

Grafické programovací prostředí RobotC	*Stručný postup (problémy řešené při práci, způsob řešení)*	*Hodnocení vyučujího (splněno/nesplněno)*
Aktivita 1 - Ruční mixér Datum:		
Aktivita 2 - Výstražný semafor se zvukovou signalizací Datum:		
Aktivita 3 - Detektor pádu pevného disku Datum:		

Projektový deník - Textově orientované programovací prostředí RobotC

Jméno:
Třída:

Programovací prostředí RobotC	*Stručný postup (problémy řešené při práci, způsob řešení)*	*Hodnocení vyučujího (splněno/nesplněno)*
Aktivita 1 - Lopatková turbína Datum:		
Aktivita 2 - Kuchyňská minutka Datum:		
Aktivita 3 - Řízení pásového transportéru Datum:		
Aktivita 4 - Železniční přejezd Datum:		
Aktivita 5 - Kreslící tabulka Datum:		

Programovací prostředí EV3	*Stručný postup (problémy řešené při práci, způsob řešení)*	*Hodnocení vyučujího (splněno/nesplněno)*
Aktivita 6 - Ruční mixér Datum:		
Aktivita 7 - Rozpoznávač barev Datum:		
Aktivita 8 - Detektor pádu pevného disku Datum:		
Aktivita 9 - Turniket Datum:		
Aktivita 10 - Bomba Datum:		
Aktivita 11 - Detektor světla Datum:		
Aktivita 12 - Stopky Datum:		