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

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

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:

Vs


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:

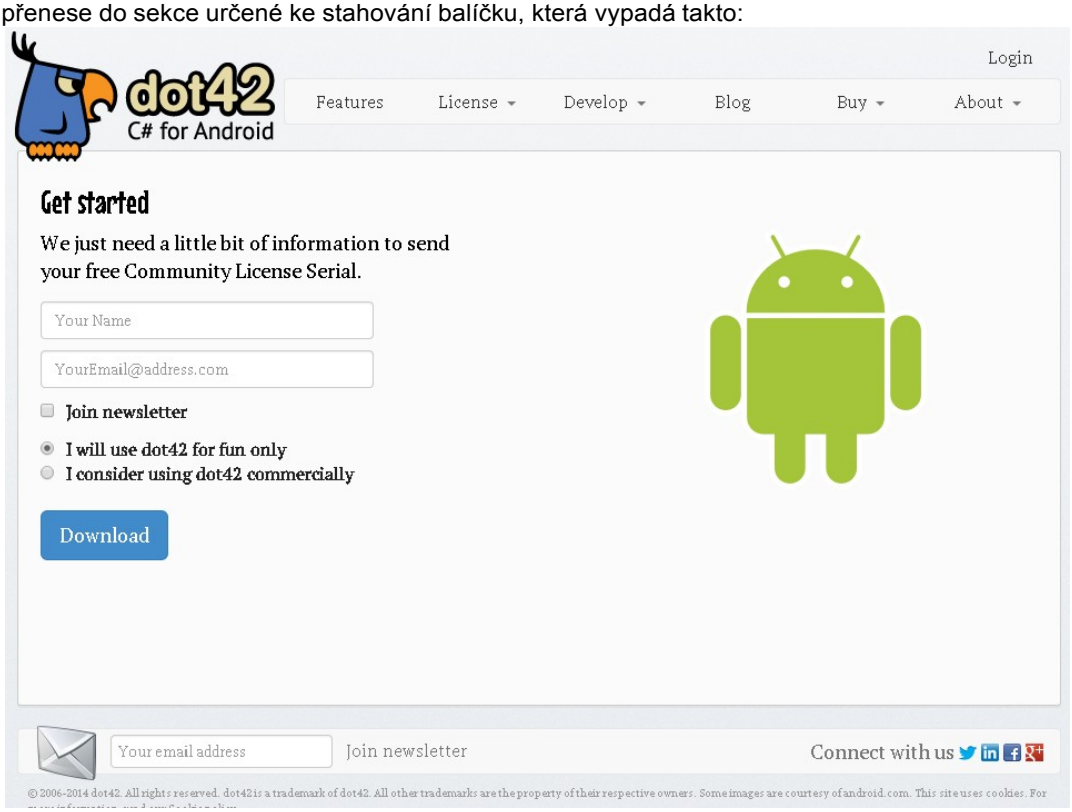
1. Zaregistrujte se a stáhněte si instalační balíček nástrojů dot48.

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

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

Postup:

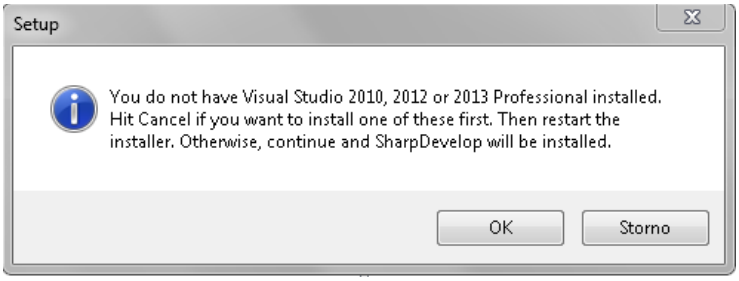
V první fázi je třeba jít na stránku projektu dot48 a zaregistrovat se proto, abychom byly schopni stáhnout jejich instalační balíček. Na adrese https://www.dot42.com/ zvolíme snadno viditelné tlačítko DOWNLOAD, které nás přenese do sekce určené ke stahování balíčku, která vypadá takto:



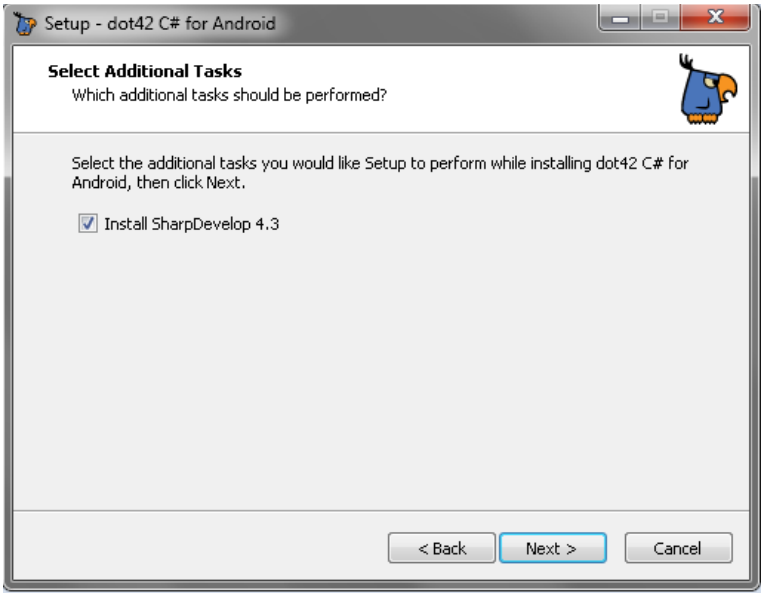
Zde je potřeba vyplnit vaše jméno a e-mailovou adresu. Dejte si pozor, abyste vyplnili adresu správně a k e-mailovému účtu měli přístup. Na tuto adresu vám totiž bude doručen aktivační klíč celého produktu. Dále je třeba zaškrnout, že nebudete používat dot42 pro komerční účely, ale pouze ke své vlastní 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.

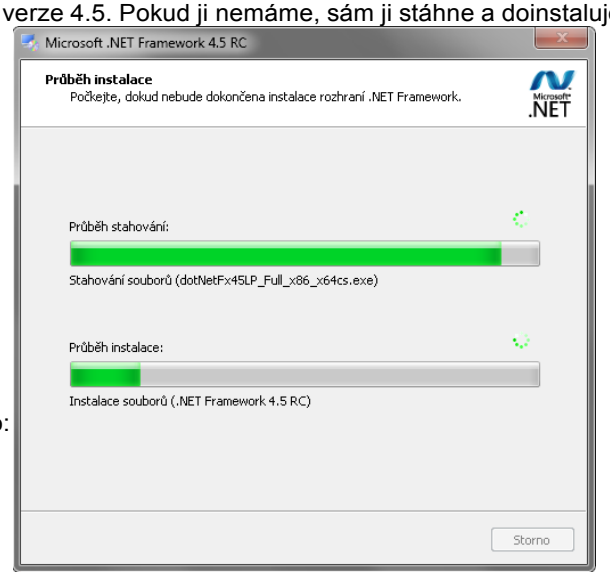
Jakmile máme balíček stažen, spustíme ho pro instalaci. Jako první vám pravděpodobně vyskočí tato hláška:



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!

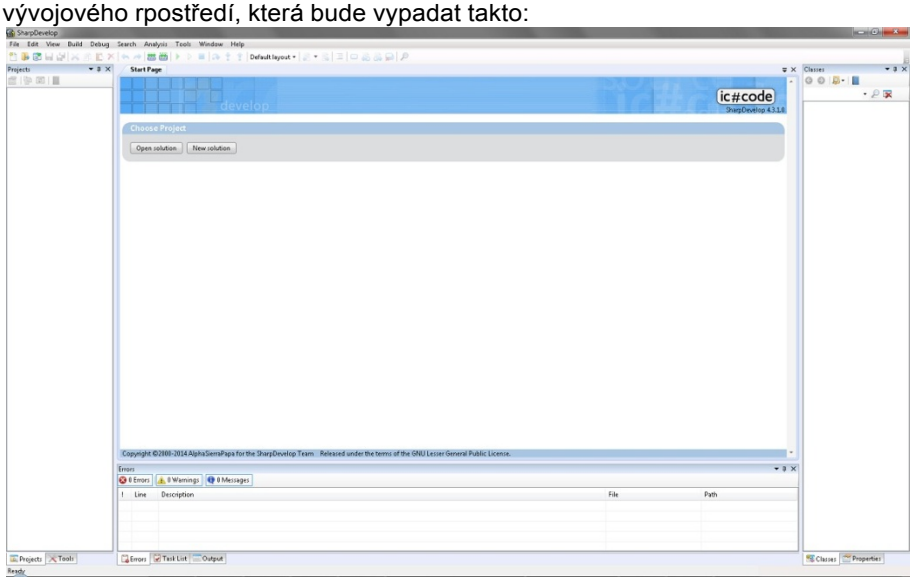
Use this serial number when you run dot42 for the first time.

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

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

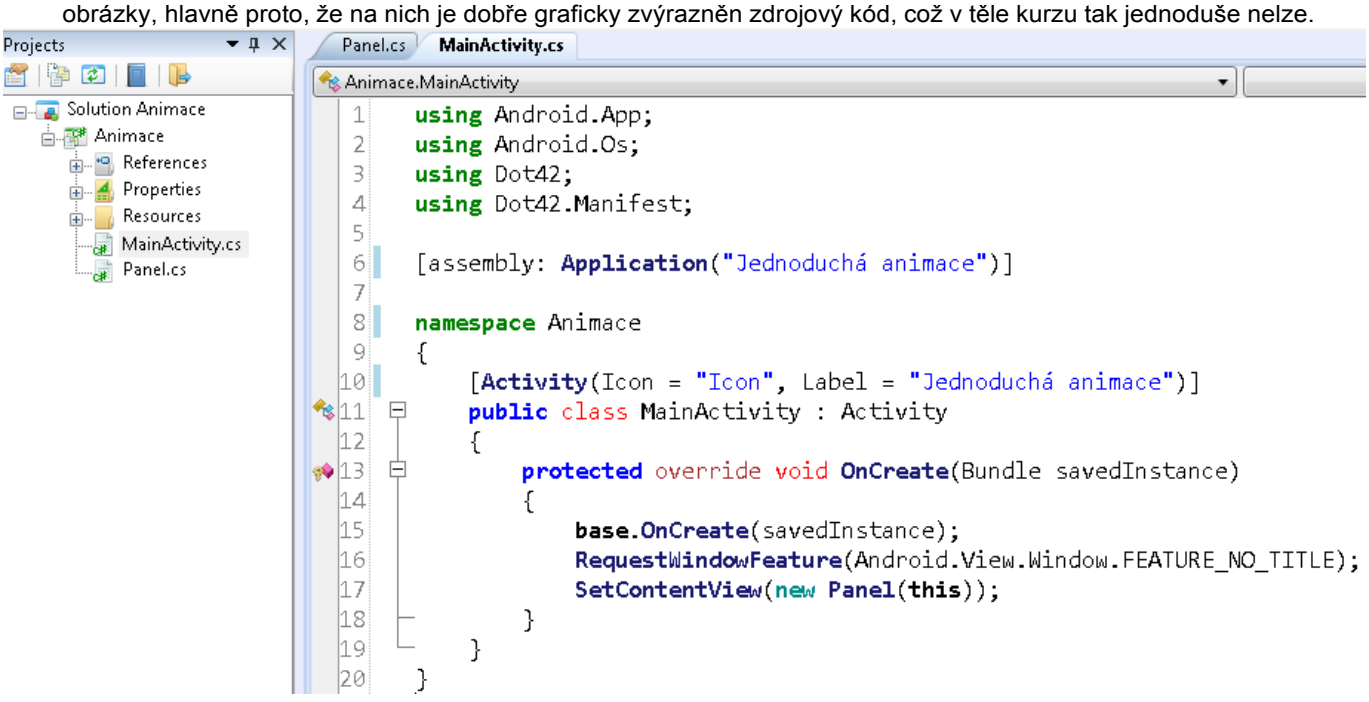
Téma	C# aplikace pro Android a práce s animacemi	
Tematický celek	Tvorba mobilní Android aplikace v jazyce C#	
Motivační rámec	Když už jsme si pohráli s klasickými ovládacími prvky, vyzkoušíme si teď vytvořit jednoduchou animaci. (Aneb je to barevné a hýbá se to!)	
Počet žáků	15	
Věk žáků	16-18	
Pomůcky	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)

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.	

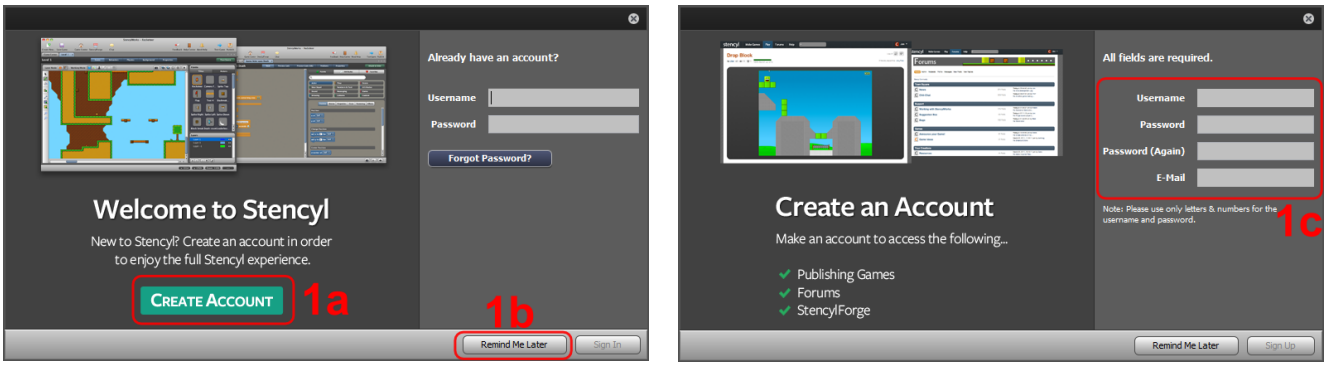
Doporučený multimediální materiál

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

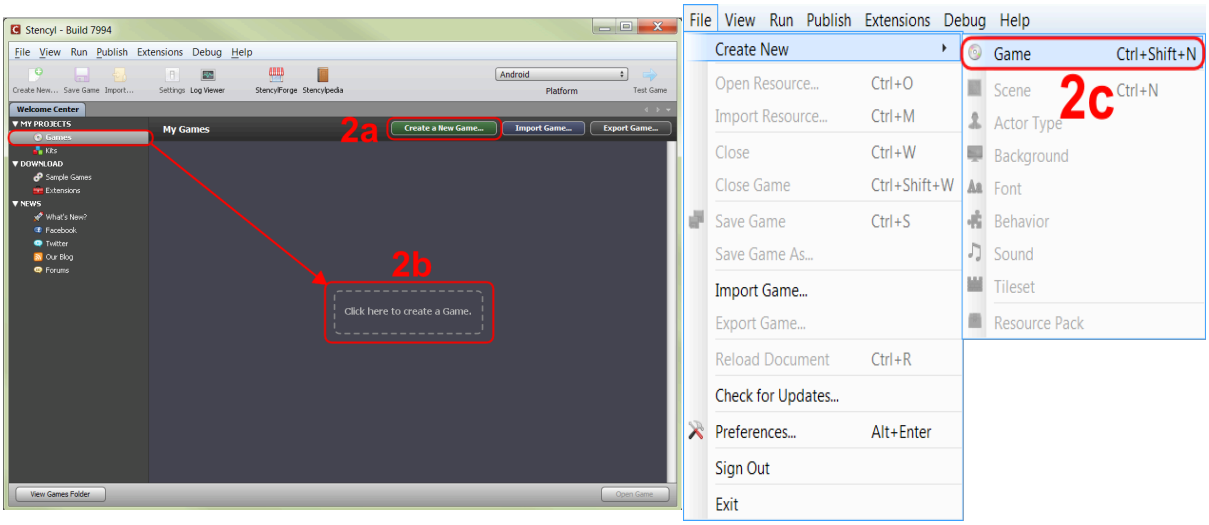
Instalace programu probíhá standardním způsobem (postupnými kroky se zobrazováním dialogových oken s umístěním instalované aplikace na lokálním disku) jako u většiny aplikací, proto komukoliv, kdo někdy nějaký program instaloval, nebude cizí ani postupinstalace tohoto programu.
Po prvním spuštění programu 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**).



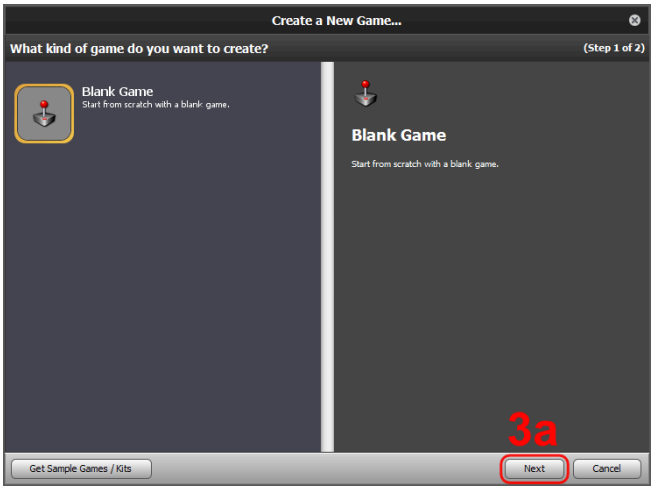
Registraci lze provést také přímo na webové stránce produktu Stencyl: www.stencyl.com/register

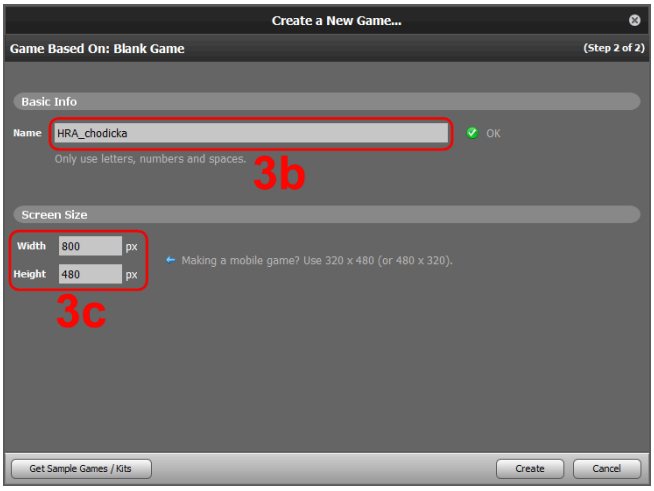
4. Vytvoření (založení) nové hry

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**).



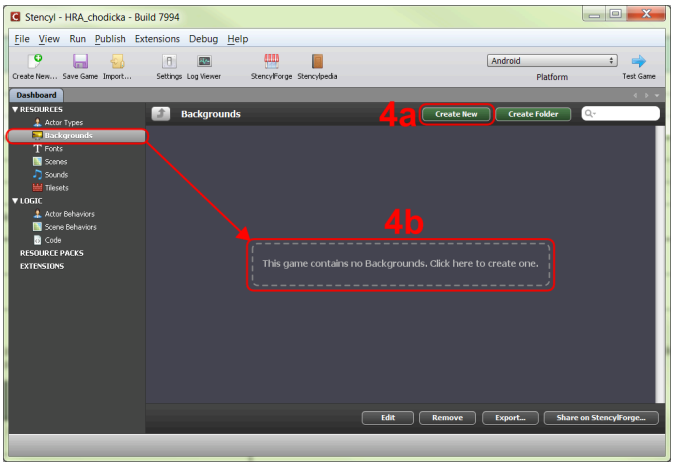
V úvodu vytvoření projektu s novou hrou je možné tlačítkem **Get Simple Games / Kits** vybrat z již hotových volně dostupnýchherních projektů, které byly realizovány komunitou Stencylvybrat nebo vytvořit prázdný projekt s hrou **Blank Game** od začátku a pokračovat (obrázek **3a**).
V dalším kroku po 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.

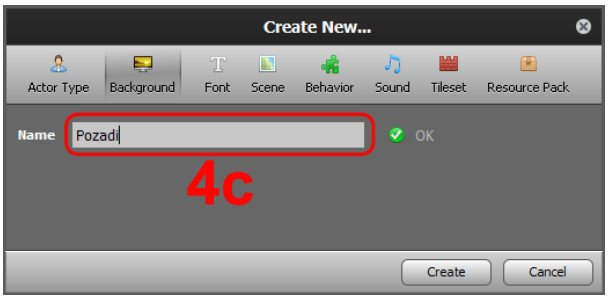




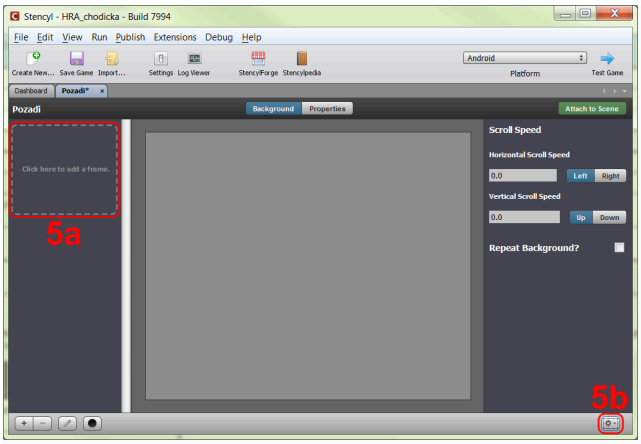
5. Příprava pozadí hry

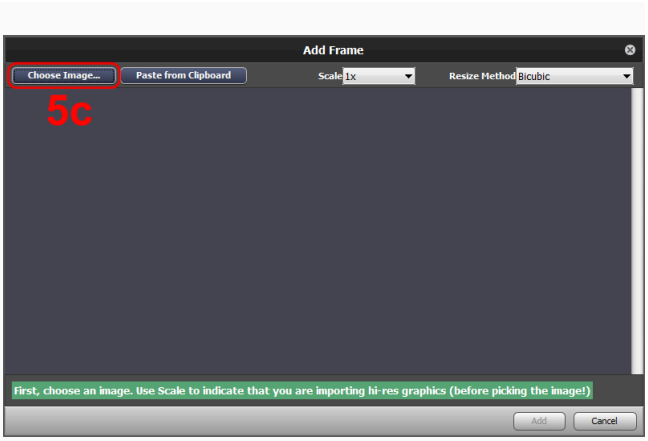
K vytvoření pozadí ve hře je nutné si připravit vhodný obrázek nebo barevnou plochu, na které dobře vyniknou (budou v kontrastu) ostatní herní prvky, jako hlavní hrdina, herní bloky (reprezentující zem, po které se bude hrdina pohybovat) a bonusy, které budehrdina sbírat.
Příprava položky reprezentující 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.





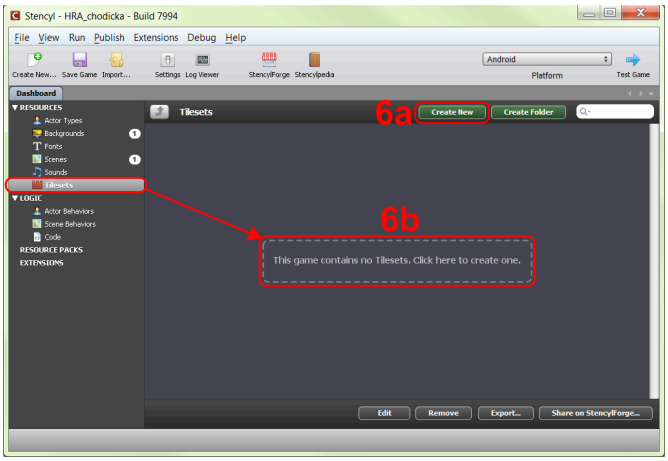
Snímek herního pozadí může být přidán kliknutím do oblasti ohraničené čárkovanou čarou v levé části pracovního okna (obrázek **5a**) nebo tlačítkem s ikonou ozubeného kola v pravém dolním rohu pracovního okna (obrázek **5b**) a následně volbou **Add Frame**. V dialogovém okně s názvem Add Frame pomocí 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**.

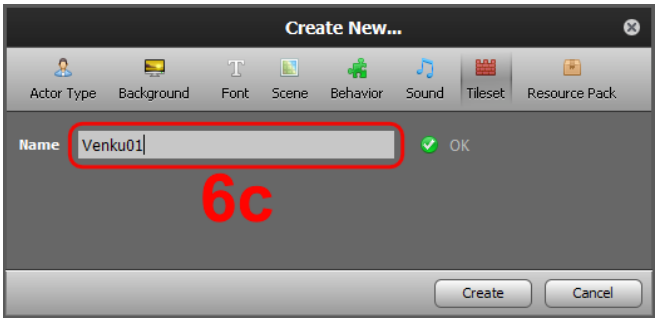




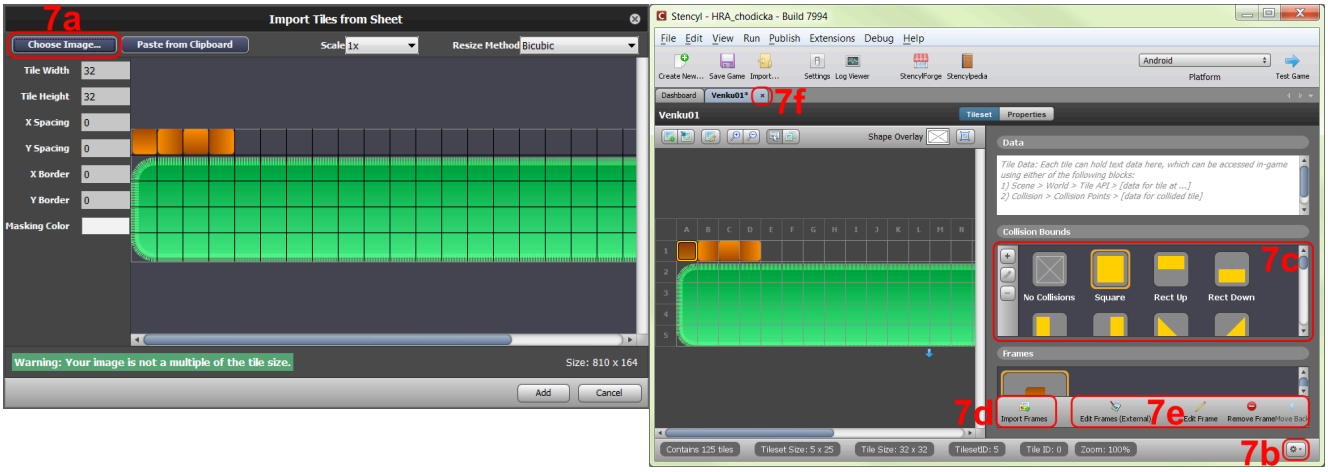
6. Realizace herního prostředí

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 bytvidět 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=xIIabLJf8To

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.