Google Web Toolkit prakticky

Jiří Semecký
jirka@google.com

Instalace GWT a podpory do Eclipse

Pro práci s GWT doporučuji použít některé ze silných Javových IDE. Existuje podpora pro IntelliJ IDEA , Eclipse i NetBeans.
Pro jednoduchost budeme nyní používat vývojové prostředí Eclipse. Pokud ho nemáte nainstalované, bude to drobné zdrženi, můžete ho ale stáhnout z adresy http://www.eclipse.org/downloads/. Vyberte verzi Eclipse IDE for Java EE Developers, která umožňuje vyvíjet webové aplikace a spouštět je v rámci aplikačního serveru.

Dalším krokem je stažení a instalace samotného Google Web Toolkit. Nejnovější verzi můžete stáhnout na adrese http://code.google.com/intl/cs/webtoolkit/download.html, vyberte verzi pro váš operační systém. Instalace se provádí prostým rozbalením archivu na požadované místo.

Dále se nám bude pro práci hodit plug-in Google Webtoolkit Tooling, který do Eclipse přidá podporu pro práci s GWT. Ten nainstalujete z Update Site http://eclipseguru.org/ podle následujícího popisu.

1. V menu Eclipse vyberte **Help** ⏲ **Software Updates...**
2. Vyberete tab **Available Software**
3. Zvolíte **Add Site...** a zadáte URL **http://eclipseguru.org/**
4. Ze seznamu vyberete přidanou update site http://eclipseguru.org/ a zaškrtnete **Google Webtoolkit (GWT) Tooling**
5. Stisknete **Install...** a pak již jen potvrzujete, na co se Eclipse zeptá

Instalace by neměla trvat více než několik málo minut

Pokud máte Google Webtoolkit Tooling nainstalovaný (Eclipse je po restartu), nastavte v **Preferences** Eclipse v záložce **GWT** do pole **GWT Home Directory** odkaz na místo, kam jste GWT nainstalovali. Druhé pole **Custom Module Template** nechte prázdné.

Úkol 1: První aplikace

Začneme s naprosto jednduchou ("hallo world") aplikací.

Na té si ukážeme základní stavební prvky GWT aplikace.
Iniciální adresářovou strukturu GWT projektu je možné vytvořit buď z příkazové řádky pomocí skriptu appliactionCreator, který je součástí GWT, nebo (pokud máte nainstalovaný plug-in pro Eclipse) přímo pomocí průvodců v Eclipse.
Popíši zde způsob vytvoření projektu v Eclipse, který rovnou vytvoří součástí nutné pro deployování

webového projektu a aplikační server (to ale zatím nebudeme používat). Použití applicationCreatoru je popsané na uvedené stránce.

1. Založíme dynamický webový projekt
   V menu Eclipse zvolte **File** - **New** - **Project...** - **Web** - **Dynamic Web Project**
   Potvrďte **Next**
2. Zadejte jméno vašeho projektu
   Nastavte **Dynamic Web Module version** na hodnotu **2.5**
   Nastavte **Configuration** na hodnotu **Google Web Toolkit Project**
   Potvrďte **Finish**

   ... v tuto chvíli je vytvořen prázdný projekt můžete si prohlédnout jeho strukturu v Project Exploreru.
3. Nyní přidáme do projektu první modul
   V menu Eclipse zvolte **File** - **New** - **GWT** - **Other** - **GWT Module**
4. Zadejte rozumný java package, ve kterém bude modul umístěn, např. **cz.fjfi.gwt.firstApp**
   Zadejte jméno modulu - jelikož to bude zároveň jméno třídy, která modul implementuje, používejte CamelCase
   Potvrďte **Finish**

Nyní je připravená vaše první aplikace, která obsahuje jeden modul. Modul v GWT je balíček, který obsahuje nastavení závislostí a odkaz na implementující třídu. Jedna stránka/aplikace může obsahovat libobolné množství modulů, zpravidla ale není potřeba aby obsahovala více než jeden! Jeden modul může obsluhovat i vzájemně nesousedící prvky stránky tak, jak je vidět ve vaší současné aplikaci.

Prohlédněte si vygenerovaný kód, důležité soubory aplikace jsou tyto:

1. Definice modulu - soubor **<package>/<jmeno_modulu>.gwt.xml**
2. HTML stránka obsahující modul - soubor **<package>/public/<jmeno_modulu>.html**
3. Implementace modulu v Javě - třída **<package>/client/<jmeno_modulu>**

Aplikaci nyní spustíme v hosted modu.

1. V menu Eclipse vyberte **Run** - **Debug Configuration...**
2. Založte novou konfiguraci GWT tak, že dvoj-kliknete na **GWT Browser** v seznamu konfigurací vlevo.
3. Vyplňte jméno, projekt a modul a ostatní volby nechte na defaultních hodnotách
   [vidíte, že je možné připojit se k již běžícímu aplikačnímu serveru, na kterém je nainstalován backend poskytující data. Pro nás bude lepší použít interní Tomcat na portu 8888]
   Tím jste založili *launching configuration*, která popisovat, jak se má vaše aplikace spouštět. Tuto konfiguraci se budete odkazovat při spouštění/debugování aplikace, ale už ji nemusíte upravovat.
4. Zvolte Debug
5. Vaše aplikace se spustí v hosted modu - objeví se dvě okna - Google Web Toolkit Develpment Shell (logovacími výpisy) a hosted mode browser (s vaší aplikací)
6. Pokud nastala nějaká chyba, odlaďte ji
7. Vyzkoušejte svou první aplikaci

Úkol 2: Ještě jednodušší než Hallo World

Prohlédněte si nyní ještě jednou soubory, které byly vygenerované - jsou to 3 soubory umístěné v adresáři **src**.
Ostatní části projektu budeme ignorovat - obsahují nastavení webového projektu, které se využije pro umístění hotové aplikace na webový server (Tomcat a pod.). Nejedná se o soubory specifické pro GWT, ale o obecné nastavení webového projektu.

Tuto jednoduchou aplikaci nyní ještě zjednodušíme tak, aby obsahovala opravdu jen nejnutnější minimum.
V HTML souboru:

1. V sekci **head** odstaňte kaskádové styly, necháme tam pouze title.
2. V sekci **body** je statický text - ten se zobrazí standardním způsobem a s naší aplikací nijak neinteraguje. Aby kód nekomplikoval, můžete ho také odstranit.
3. Dále následuje tabulka, která obsahuje dvě buňky s id slot1 a slot 2. Předěláme aplikaci tak, že bude

používat pouze 1 slot a budeme ho reprezentovat jako DIV. Celou tabulku tedy nahraďte elementem:

Pokud bych chtěl stránku maximálně zjednodušit, mohu z ní nechat pouze toto:

```
<html>
    <body>
        <script language="javascript"
src="gwtintro.first.FirstModule.nocache.js"></script>
        <iframe id="__gwt_historyFrame"
style="width:0;height:0;border:0"></iframe>
        <div id="slot1" />
    </body>
</html>
```

Více není pro GWT potřeba. Scirpt se odkazuje na naši zkompilovanou aplikaci. Řádka s **iframe** zajišťuje správnou funkčnost historie (tlačitek Forward a Back) v MS IE, bez něj by aplikace běžela také. Podlední prvek - div definuje prostor, kde naše aplikace poběží. Když nyní aplikaci spustíte, skončí chybou. Proč?

V původní aplikaci byly dva prvky a jejich layout byl řešen HTML tabulkou. Nyní jsme nechali pouze jeden slot, změňte Java aplikaci tak, aby se odkazovala pouze na něj a layout vytvořte v Javě. Přečtěte si HTML Host Pages, kde je popsáno mapování HTML do Javy.
Prostudujte si nyní, z jakých prvků můžete tvořit uživatelského rozhraní. Pro layout z našeho příkladu se vám bude hodit třída HorizontalPanel.

Úkol 3: Uživatelské rozhraní, debuggování a kompilace do web modu

Navrhněte a implementujte uživatelské rozhraní, které bude obsahovat mimo jiné editační řádku pro zadání věty, seznam (ListBox) vět a případně další prvky (tlačítka a pod.).
Uživatel bude moci vloženou větu přidat do seznamu. Dále bude možné seznam vyprázdnit, případně i smazat jednotlivou řádku.
Pro implementaci rozhraní použijte prvky popsané zde. Bude se vám např. hodit metoda `ListBox.setVisibleItemCount(int)`.

Aplikaci odlaďte pomocí debuggeru v Eclipse. Ten se chová tak, jako by se jednalo o normální Java aplikaci. Ono se totiž v hosted modu ve skutečnosti o Java aplikaci jedná.
V **GWT Browseru**, kde běží vaše aplikace v hosted modu je tlačítku **Combile/Browse**, kterým aplikací zkompilujete do JavaScriptu. Tím se pro každý podporovaný browser vytvoří jeden JavaScriptový soubor s vaším přeloženým kódem. Soubory si můžete prohlédnout v adresáři **`bin.gwt`**. (v **Package Exploreru** není vidět, ale můžete použít view **Navigator**).

Úkol 4: RPC - komunikace se serverem

V této části naimplementujeme komunikaci mezi klientem a serverem. Klient GWT se může připojovat na libovolné backendy, napsané v nejrůznějších programovacích jazycích. Máte k dispozici podporu na parsování odpovědi v JSON i XML. Pokud se ale rozhodnete použít interní mechanismus GWT RPC pro komunikaci se serverm psaným v Javě, budete mít práci hodně zjednodušenou, komunikace probíhá takřka transparentně. Serverová část je implementována jako servlet běžící na aplikačním serveru (např. Tomcat).

Projděte si tyto stránky (http://code.google.com/intl/cs/docreader/#p=google-web-toolkit-doc-1-5&s=google-web-toolkit-doc-1-5&t=DevGuidePlumbingDiagram a http://code.google.com/intl/cs/docreader/#p=google-web-toolkit-doc-1-5&s=google-web-toolkit-doc-1-5&t=DevGuideCreatingServices), abyste měli představu, jak GWT RPC funguje.
V Eclipse můžete vytvořit servis pomocí wizardu:

1. Založíme dynamický webový projekt
   V menu Eclipse zvolte **File** - **New** - **Other...** - **GWT** - **GWT Remote Service**
   Potvrďte **Next**
2. V poli **Module Location** vyberte svůj GWT modul
   Nastavte jméno servisu, např. BabelService (brzy se dozvíte proč :)) Pro pojmenování modulu opět použijte *CamelCase*.
   Jako Service URI nastavte identifikátor, pod kterým bude na webovém serveru servlet přístupný.

Může to být cokoli, co může bát součást URL, např. bableService.
Potvrďte **Finish**

Vytvořili jste kostru Remote Servisu, která obsahuje:

- Interface pro tento service (**BabelService.java**), zatím prázdný. Tady budete deklarovat metody, které chce, aby mohl klient na serveru spouštět.
- Asynchroní interface (**BabelServiceAsync.java**), který obsahuje přesně stejné metody jako ten předchozí. Jediný rozdíl je v tom, že jsou psány asynchronně, tedy nevrací žádnou hodnotu (mají návratový typ void), ale zato mají navíc parametr s asynchroním callbackem - to je třída, která se zavolá, když přijde ze serveru odpověď.
  V GWT projektu je potřeba, abyste tyto dva interfacy udržovaly tak, že si budou odpovídat. Pokud ale používáte plug-in pro Eclipse, nemusíte se o to starat a tento - asynchroní interface - nebudete upravovat, sám se bude měnit podle změn v prvním interface.
- Serverová implementace interface (**BabelServiceImpl.java** v balíku **server**) - tam umístíte kód, který tento servis implementuje. Je to třída, která dědí po RemoteServiceServlet, což je GWT implementace servletu (zajišťuje serializaci dat).

Ještě je třeba aplikačnímu serveru říct, na jakém URL má servlet zpřístupnit. To uděláte přidáním elementu servlet do definice modulu (soubor <modul>.gwt.xml).

```
 <servlet path="/babelService"
class="cz.fjfi.gwt.workshop.server.BabelServiceImpl" />
```

Tím jsme řekli aplikačnímu serveru Tomcat (ten je použitý v hosted modu), že má spřístupnit náš servlet pomoví URI babelService. Tohle by asi mohl dělat rovnou ten wizard, že?

Přidejte nyní do synchroního interface metodu pro kódování řetězce:

```
String koduj(String str);
```

Zkontrolujte, že se metody objevila i v asynchroním interface. Metodu naimplementujte tak, aby obrátila vypsala slova v obráceném pořadí (nebo si vymyslete jiné kódování).

Nyní se podíváme na to, jak metodu zavolat. Nejprve si vytvoříme třídu implementující asynchroní interface - ta reprezentuje náš servis na klientu. K tomu použijeme metodu create třídy GWT. Uvědomte si, že tento interface jsme ještě neimplementovali. Ani nemusíme, udělá to za nás GWT.

```
private BabelServiceAsync service;
...
service = GWT.create(BabelService.class);
```

Všimněte si, že GWT.create() předáváte třídu synchroního servisu, ale ta vám vrátí asynchroní verzi. Nyní asynchroně zavoláme metodu tak, že předáme asynchroní callback. Při zavolání bude běh programu pokračovat normálně dál a až přijde od serveru odpověď, zavolá se odpovídající metody callbacku:

```
service.koduj(text.getText(), new AsyncCallback<String>() {

    // došlo k chybě
    public void onFailure(Throwable res) {
        Window.alert("Při volání metody na serveru došlo k chybě");
    }

    // komunikace proběhla správně
    public void onSuccess(String res) {
        list.addItem(res);
    }

});
```

Pro odladění komunikace použijte debugger, lze stejně debuggovat serverovou jako klientskou část.

Úkol 5: Překladač - kombinace s dalšími knihovnami

Dosud jsme na serveru dělala velmi jednoduché zpracování. Upřímně řečeno by šlo něco takového jednoduše napsat i na klientu. V tomto úkolu si vyzkoušíme nějaký mashup - propojíme dohromady dvě zajímavé technologie. Přidáme tlačítko na přidání řádky v jiném jazyce. Navíc to UI dejte dva komboboxy, jeden pro určení zdrojového jazyka a druhý pro cílový jazyk (místo názvů jazyků můžete použít jejich kódy -

"cs", "en", "de", "es", "ru", "it" apod.). Když uživatel stiskne tlačítko, do seznamu vět se nevloží napsaná věta, ale její překlad z/do určeného jazyka? V průběhu semináře se podíváme na příklad.

Vlastní překlad nepište sami, to byste během semináře nestihli, místo toho můžete použít knihovnu google-api-translate-java.
Pokud úkol nestihnete během semináře dopsat, budu rád, když mi pošlete link na běžící aplikaci.

A co dál? Od verze 1.5 nabízí GWT funkcionalitu ImageBundle - je to mechanismus, který za vás spojí řadu obrázků (např. ikon) do jednoho obrázku a ty pak můžete na klientu používat. Výhoda je ta, že se fyzicky přenáší pouze jeden obrázek, což je jednak rychlejší (menší data) a jednak to zabrání nepříjemnému efektu postupného načítání obrázků. Pro vás jako programátora se to přitom chová tak, jakobyste pracovali s každým obrázkem zvlášť. A co s tím? Bylo by např. pěkné uživatelské rozhraní jednoduše vylepšit tím, že do seznamu jazyků přidáte ikony s vlajkou daného státu.

Závěrem

Doufám, že vám letmý úvod do GWT byl užitečný a že se vám práce s GWT líbí. Nastínili jsme si základní vlastnosti GWT - tvorbu uživatelského rozhraní, debuggování v Javovém vývojovém prostředí, komunikace se serverem a dotkli se dalších částí. Tím ale rozhodně množina funkcí a vlastností GWT nekončí. Pokud budete mít zájem, tady je několik odkazů, které se vám monou při psaní GWT aplikací hodit.

- Projekt GWT - http://code.google.com/intl/cs/webtoolkit/
- Diskusní skupina uživatelů GWT - http://groups.google.com/group/Google-Web-Toolkit
- Příklady použití GWT - http://code.google.com/intl/cs/webtoolkit/examples/
- GWT tutorial - http://code.google.com/intl/cs/docreader/#p=google-web-toolkit-doc-1-5&s=google-web-toolkit-doc-1-5&t=GettingStarted
- GWT Ext - knihovna pro pokročilejší uživatelské rozhraní - http://code.google.com/p/gwt-ext/
- Slidy k přednášce o GWT - https://docs.google.com/a/semecky.cz/Presentation?docid=dg92rh9m_88hfwf24hc&hl=cs