Webové aplikace 2

GWT

Martin Klíma

GWT – co to je?

Google Web Toolkig

- Překladač z jazyka Java do JavaScript + HTML
- Sada JavaScript a Java skriptů / tříd
- Vývojové prostředí – SDK
- Integrace s IDE – Eclipse, Netbeans, …

Myšlenka:

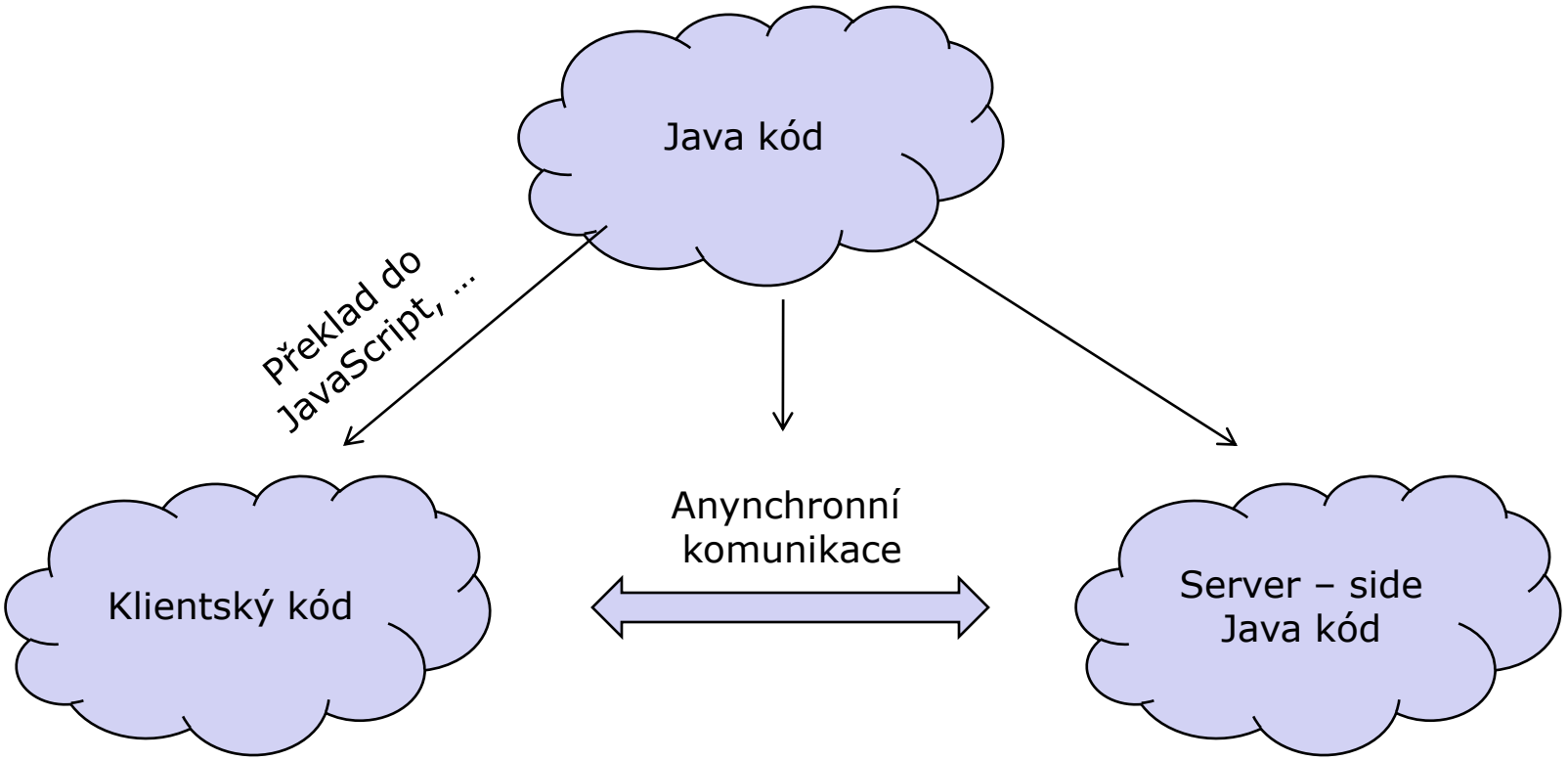
Programujme vše v jednom jazyce – Java

O klienta se postará překladač

Psaní uživatelského rozhraní podobně jako v Java Swing

Psaní „klasických“ ovladačů událostí

Architektura – AJAX v hlavní roli



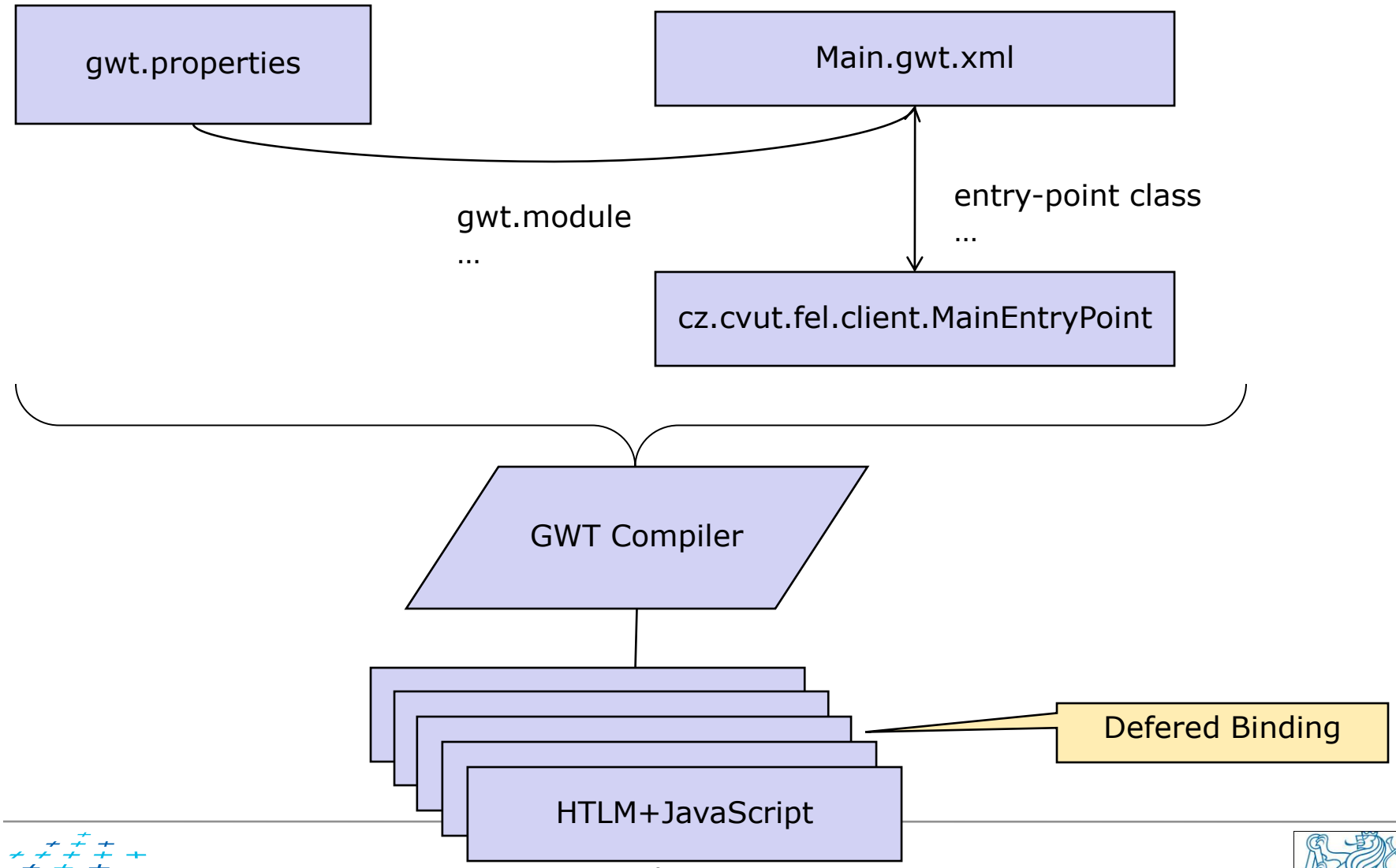
Ukázka

dokonce živá :-)

GWT Moduly

- Projekt se v GWT dělí na moduly
- Modul je ohraničená funkcionalita
- Mají jmennou konvenci jako v Javě
- Mají deskriptor s koncovkou .gwt.xml
  - Zděděné moduly
  - Entry-point class
  - Zdrojové cesty
    - jen třídy v těchto cestách budou přeloženy do JavaScriptu. Pozor, toto je častý zdroj chyb – nepochopení.
  - Veřejné cesty
  - Externí JavaScript
  - CSS
  - Servlety, další „property“

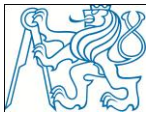
Jak to funguje?



GWT Properties

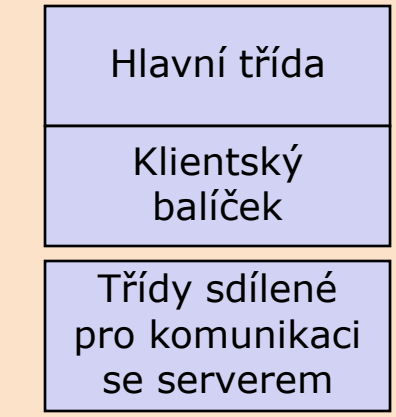
```
# The name of the module to compile
gwt.module=cz.cvut.fel.dama

# Folder within the web app context path where the output
# of the GWT module compilation will be stored.
# This setting is only used for GWT 1.5. For newer versions please use
# the rename-to attribute in the GWT module file (.gwt.xml).
gwt.output.dir=/cz.cvut.fel.dama

# Script output style: OBF[USCATED], PRETTY, or DETAILED
gwt.compiler.output.style=OBF

# Additional JVM arguments for the GWT compiler
gwt.compiler.jvmargs=-Xmx256M

# Specifies the number of local workers to use whe compiling permutations and module(s)
gwt.compiler.local.workers=1

# The level of logging detail: ERROR, WARN, INFO, TRACE, DEBUG,
gwt.compiler.logLevel=WARN
```

```
# Script output style: OBF[USCATED], PRETTY, or DETAILED
gwt.shell.output.style=OBF

# The level of logging detail: ERROR, WARN, INFO, TRACE, DEBUG,
gwt.shell.logLevel=WARN

# Additional JVM arguments for the GWT shell/GWT hosted mode (GWT 1.6)
# Add -d32 here and use at least GWT 1.7.1 to debug on a Mac
# (32-bit JRE is required by GWT for debugging)
gwt.shell.jvmargs=-Xmx256M

# GWT version: 1.5,1.6,1.7 or 2.0
gwt.version=2.0

# GWT 2.0 only
# Specifies the TCP port for the code server
gwt.shell.code.server.port=9997

# GWT 2.0 only
# Specifies the TCP port for the embedded web server
gwt.shell.port=8888
```

GWT - module entry point

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE module PUBLIC "-//Google Inc.//DTD Google Web Toolkit 1.7.0//EN"
"http://google-web-toolkit.googlecode.com/svn/tags/1.7.0/distro-source/core/src/gwt-module.dtd">

<module>
  <inherits name="com.google.gwt.user.User"/>
  <inherits name='com.google.gwt.user.theme.standard.Standard'/>

  <!-- Inherit the default GWT style sheet. You can change -->
  <!-- the theme of your GWT application by uncommenting -->
  <!-- any one of the following lines. -->
  <!-- <inherits name='com.google.gwt.user.theme.standard.Standard'/> -->
  <!-- <inherits name="com.google.gwt.user.theme.chrome.Chrome"/> -->
  <!-- <inherits name="com.google.gwt.user.theme.dark.Dark"/> -->

  <entry-point class="cz.cvut.fel.client.damaEntryPoint"/>

  <source path="client"/>
  <source path="shared"/>

  <!-- Do not define servlets here, use web.xml -->
</module>
```



Zavádění kódu

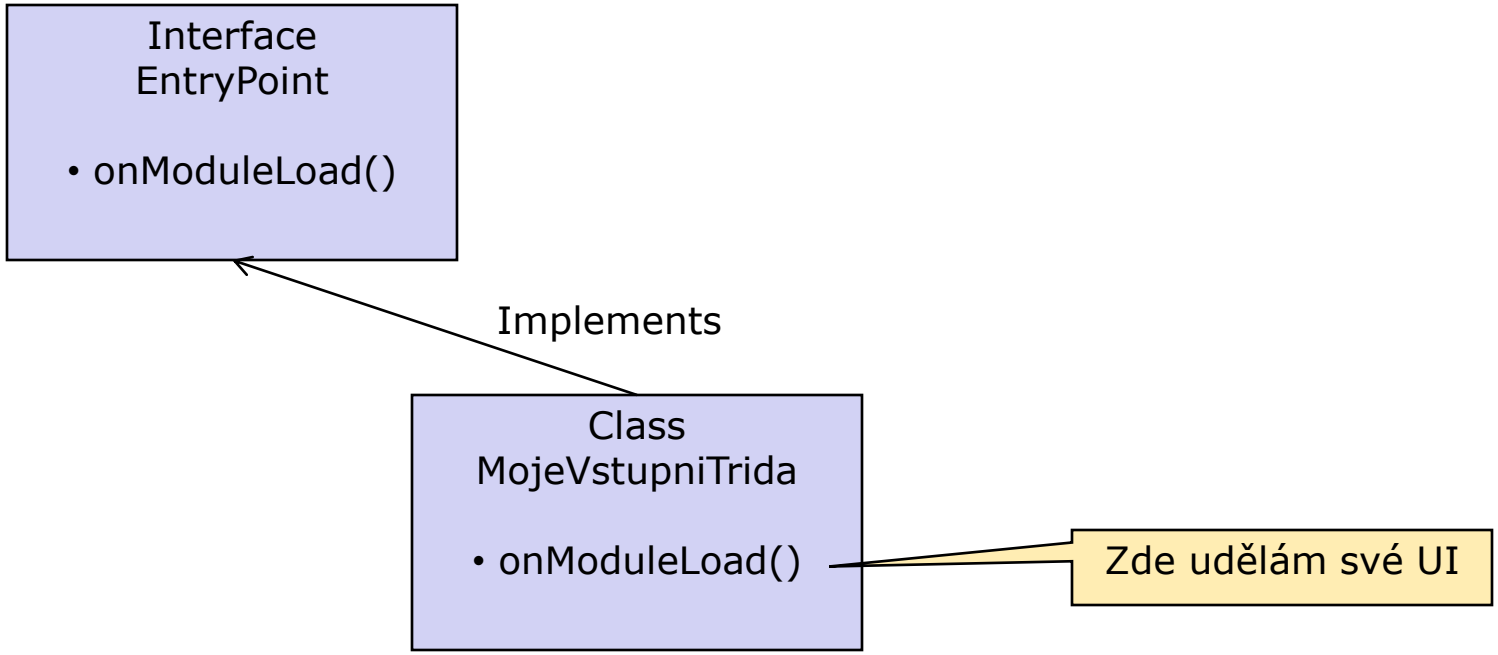
1. Prohlížeč nahraje zaváděcí HTML stránku
2. Narazí na značku *<script src="<Module Name>.nocache.js">* , nahraje JavaScript
3. V JavaScriptu je obsaženo rozhodování o derefed binding, na základě kterého se nahraje příslušná verze .cache.html
4. .cache.html vytvoří skrytý <iframe> a nahraje se do něj
5. po nahrání .cache.html se spustí vlastní kód aplikace

Zaváděcí HTML stránka

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
   <link type="text/css" rel="stylesheet" href="dama.css">
    <title>Dáma na webu</title>
    <!--  Nahrání klientské logiky v javascriptu -->
    <script type="text/javascript" language="javascript" src="cz.cvut.fel.dama/cz.cvut.fel.dama.nocache.js"></script>
  </head>
 <body>
    <!-- iframe pro podporu historie -->
    <iframe src="javascript:"" id="__gwt_historyFrame" tabIndex='-1'
style="position:absolute;width:0;height:0;border:0"></iframe>

    <h1>Webová dáma</h1>
    <div id="ovladani"></div>
    <div id="sachovnice"></div>
  </body>
</html>
```

Jednoduchý začátek – klientský kód

- Vstupní bod do UI je třída implementující EntryPoint



Jednoduchá implementace

```
package cz.cvut.fel.client;

import com.google.gwt.core.client.EntryPoint;
import com.google.gwt.user.client.ui.Button;
import com.google.gwt.user.client.ui.Label;
import com.google.gwt.user.client.ui.RootPanel;
import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;

public class MainEntryPoint implements EntryPoint {

   public MainEntryPoint() {    }

   public void onModuleLoad() {
      final Label label = new Label("Hello, GWT!!!");
      final Button button = new Button("Click me!");

      button.addClickHandler(new ClickHandler() {
         public void onClick(ClickEvent event) {
            label.setVisible(!label.isVisible());
         }
      });
      RootPanel.get().add(button);
      RootPanel.get().add(label);
   }
}
```

Kód velmi podobný Swing

Manipulace s UI čistě na straně klienta

UŽIVATELSKÉ ROZHRANÍ

UI podobné jiným Java technikám (Swing)

RootPanel – do tohoto kontejneru se vkládá vše ostatní

FlowPanel – jednoduchý ve stránce

HTMLPanel – staví HTML strukturu

FormPanel – formulář

ScrollPanel – má posuvník

PopupPanel, DialogBox - dialogy

Grid, FlexTable – tabulky

LayoutPanel, DockLayoutPanel, SplitLayoutPanel, StackLayoutPanel, TabLayoutPanel,

Více info zde:
http://code.google.com/intl/cs/webtoolkit/doc/latest/DevGuideUiPanels.html

POZOR

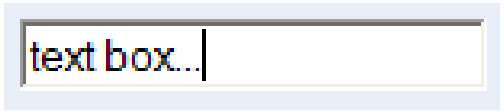
POZOR! POZOR! POZOR!

GWT layout funguje správně jen ve standardním režimu prohlížeče

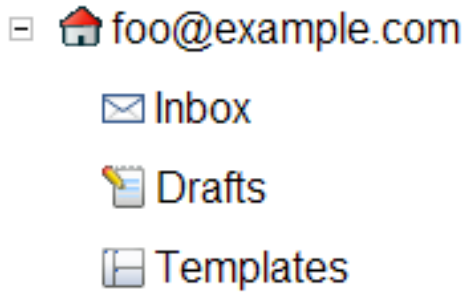
Další prvky

- Button

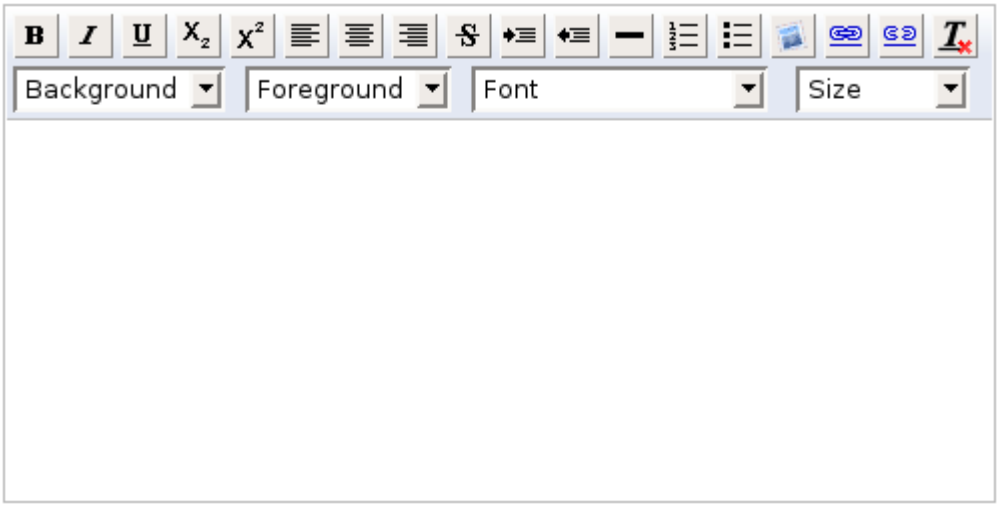
- TextBox



- Tree



- RichTextArea



Události a jejich odchycení

```
public void anonClickHandlerExample() {
      Button b = new Button("Click Me");
      b.addClickHandler(new ClickHandler() {

         public void onClick(ClickEvent event) {
            // handle the click event
         }
      });
   }
```

Anonymní vnitřní třída

pozor – předávání parametrů z vnější třídy – klíčové slovo ***final***

Jiný způsob

```
class MojeTrida implements ClickHandler {
   public void onClick(ClickEvent event) {
      // tady si odchytnu udalost a udelam s ni co potrebuji
   }
}
```

KOMUNIKACE SE SERVEREM

GWT - RPC

- RPC = Remote Procedure Call
  - z klientského kódu zavoláme serverový kód
  - Klient – JavaScript
  - Server – Servlet
  - Komunikace pomocí Ajax
  - Prostředníkem komunikace jsou serializovatelné objekty
    - *char*, *byte*, *short*, *int*, *long*, *boolean*, *float*, *double*
    - *String*, *Date, Character*, *Byte*, *Short*, *Integer*, *Long*, *Boolean*, *Float*, *Double*
    - pole serializovatelných hodnot
    - serializovatelná třída (definovaná uživatelem) (typu *IsSerializable* nebo *java.io.Serializable*)
    - třída, která má alespoň jednu serializovatelnou podtřídu

GWT – RPC pokračování

co je ve hře:

- Synchronní rozhraní
- Asynchronní rozhraní
- Implementace asychnronního rozhraní na serveru
- Volání asynchronního automaticky vygenerovaného kódu klientem
- Implementace synchronního rozhraní serverem

GWT - RPC

Klient

Server

GWT

• PozpatkuServiceAsync create (Class trida)

Synchronní rozhraní

• String pozpatku(string s)

RemoteServiceServlet

• processCall(String s)
•...

Klientský kód

```
final PozpatkuServiceAsync pozpatkuServiceAsync =
        GWT.create(PozpatkuService.class);

 pozpatkuServiceAsync.pozpatku(textbox.getText(), new
AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                // udelej neco pri chybe
            }
            public void onSuccess(String result) {
               // udelame neco s vracenym retezcem
            }
        });
```

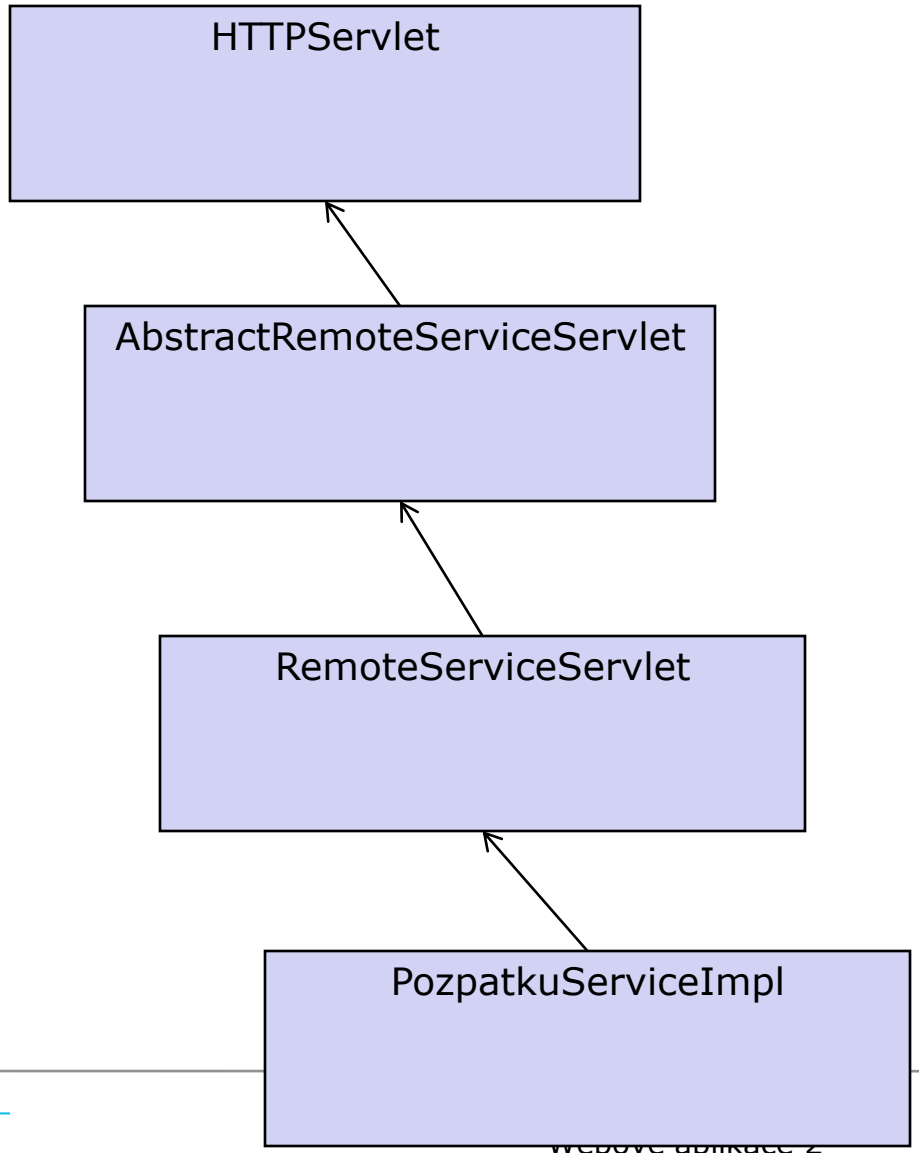
Serverový kód

```
public class PozpatkuServiceImpl extends
RemoteServiceServlet
 implements PozpatkuService {

   public String pozpatku(String s) {
      StringBuffer reverse =
       new StringBuffer(s.length());
      for (int i = (s.length() - 1); i >= 0; i--) {
         reverse.append(s.charAt(i));
      }
      return reverse.toString();
   }
}
```

Sessions

- U Ajax aplikací se stav aplikace ukládá u klienta.
- Problém s obnovením celé stránky.
- Potřeba udržování stavu i na serveru

Sessions


HTTPServlet
AbstractRemoteServiceServlet
RemoteServiceServlet
PozpatkuServiceImpl

Práce se session na straně serveru

```
public class PozpatkuServiceImpl extends RemoteServiceServlet implements PozpatkuService {

   public String pozpatku(String s) {
      StringBuffer reverse = new StringBuffer(s.length());
      for (int i = (s.length() - 1); i >= 0; i--) {
         reverse.append(s.charAt(i));
      }
      String toReturn = reverse.toString();
      // ulozim si posledni string do session
      getThreadLocalRequest().getSession().setAttribute("lastString", toReturn);
      return toReturn;
   }

   public String getLast() {
      return (String)getThreadLocalRequest().getSession().getAttribute("lastString");
   }
}
```

Nastavení session

Čtení session

GWT a historie v prohlížeči

- Problém: Ajax aplikace negenerují historii viditelnou pro prohlížeč
- Řešení v GWT: iframe, ve kterém se historie bude schovávat
- Jak to funguje:
  1. Do HTML se vloží iframe
  2. Na základě události X si do historie aktivně zapíšeme nějaký údaj (String)
  3. Na událost posunu v historii reaguji tak, že si z historie „vytáhnu“ příslušný string a podle něj nastavím aplikaci do příslušného stavu.

Do HTML se vloží iframe

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <meta name='gwt:module' content='cz.cvut.fel.Main=cz.cvut.fel.Main'>
    <title>Main</title>
  </head>
  <body>
    <script type="text/javascript"  src="cz.cvut.fel.Main/cz.cvut.fel.Main.nocache.js"></script>
    <!-- include this if you want history support -->
    <iframe id="__gwt_historyFrame" style="width:0;height:0;border:0"></iframe>
  </body>
</html>
```

Na základě události X si do historie aktivně zapíšeme nějaký údaj (String)

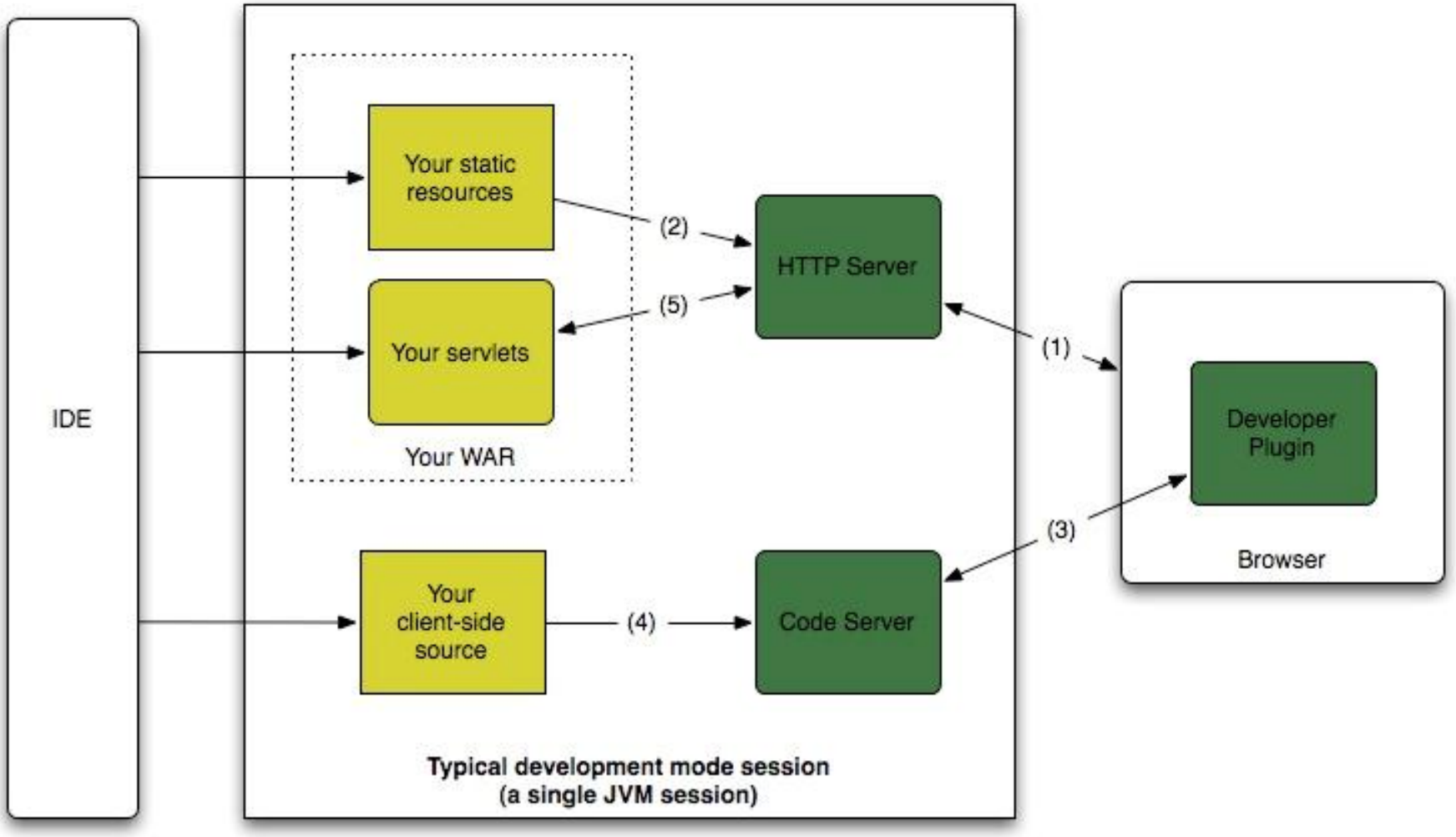
```
textbox.addValueChangeHandler(new ValueChangeHandler<String>() {

        public void onValueChange(ValueChangeEvent<String> event) {
            History.newItem("text_" + event.getValue());
        }
    });
```

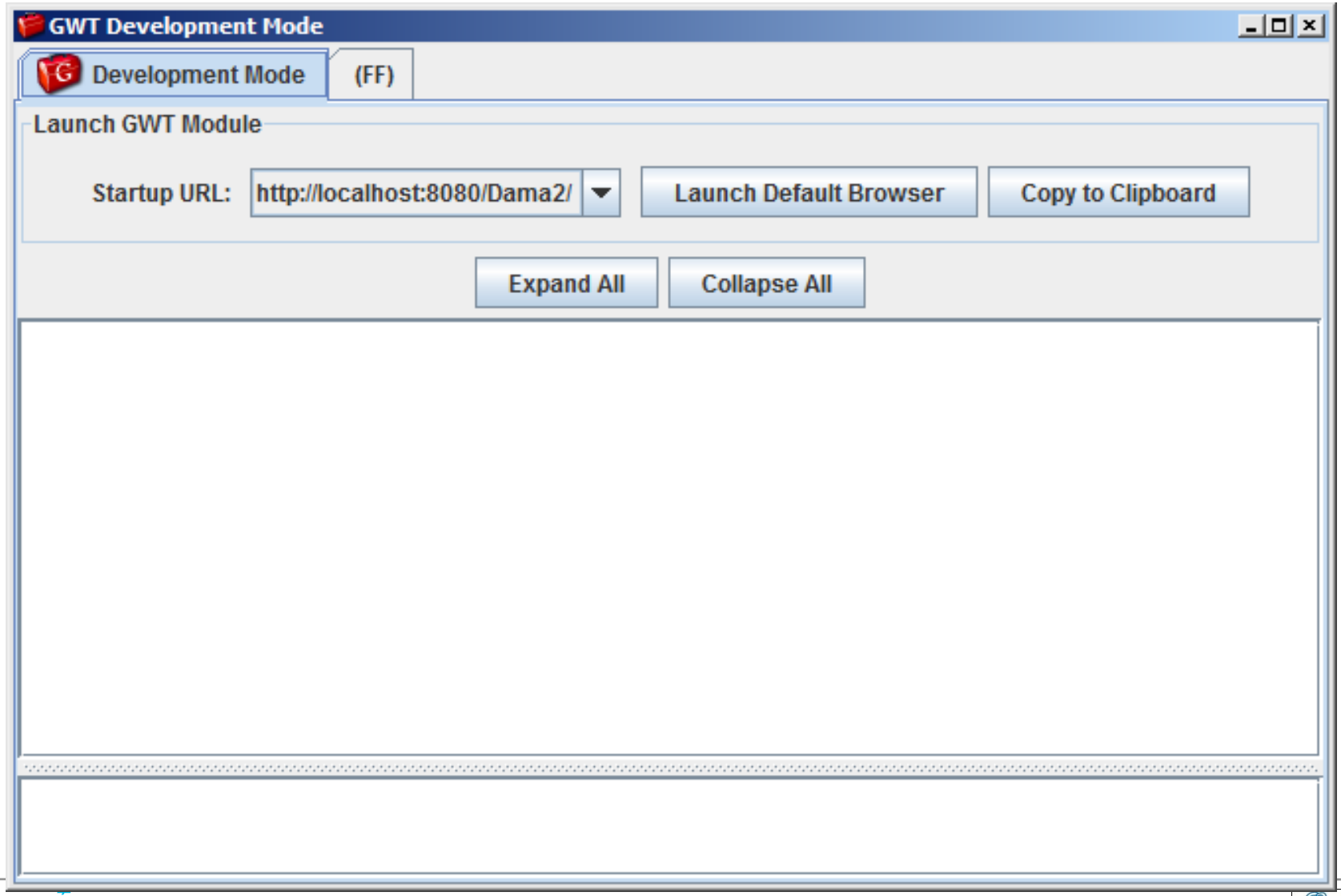
Na událost posunu v historii reaguji tak, že si z historie „vytáhnu“ příslušný string a podle něj nastavím aplikaci do příslušného stavu

```
History.addValueChangeHandler(new ValueChangeHandler<String>() {

        public void onValueChange(ValueChangeEvent<String> event) {
            String historyToken = event.getValue();

            // Parse the history token

            if (historyToken.substring(0, 5).equals("text_")) {
                String historyString =
                        historyToken.substring(5, historyToken.length());
                // Nastavime historickou hodnotu
                textbox.setText("History: "+historyString);
            } else {
                textbox.setText("");
            }
        }
    });
```

KOMPILACE A DEBUGING

Debugování ve vývojovém módu (Development mode)



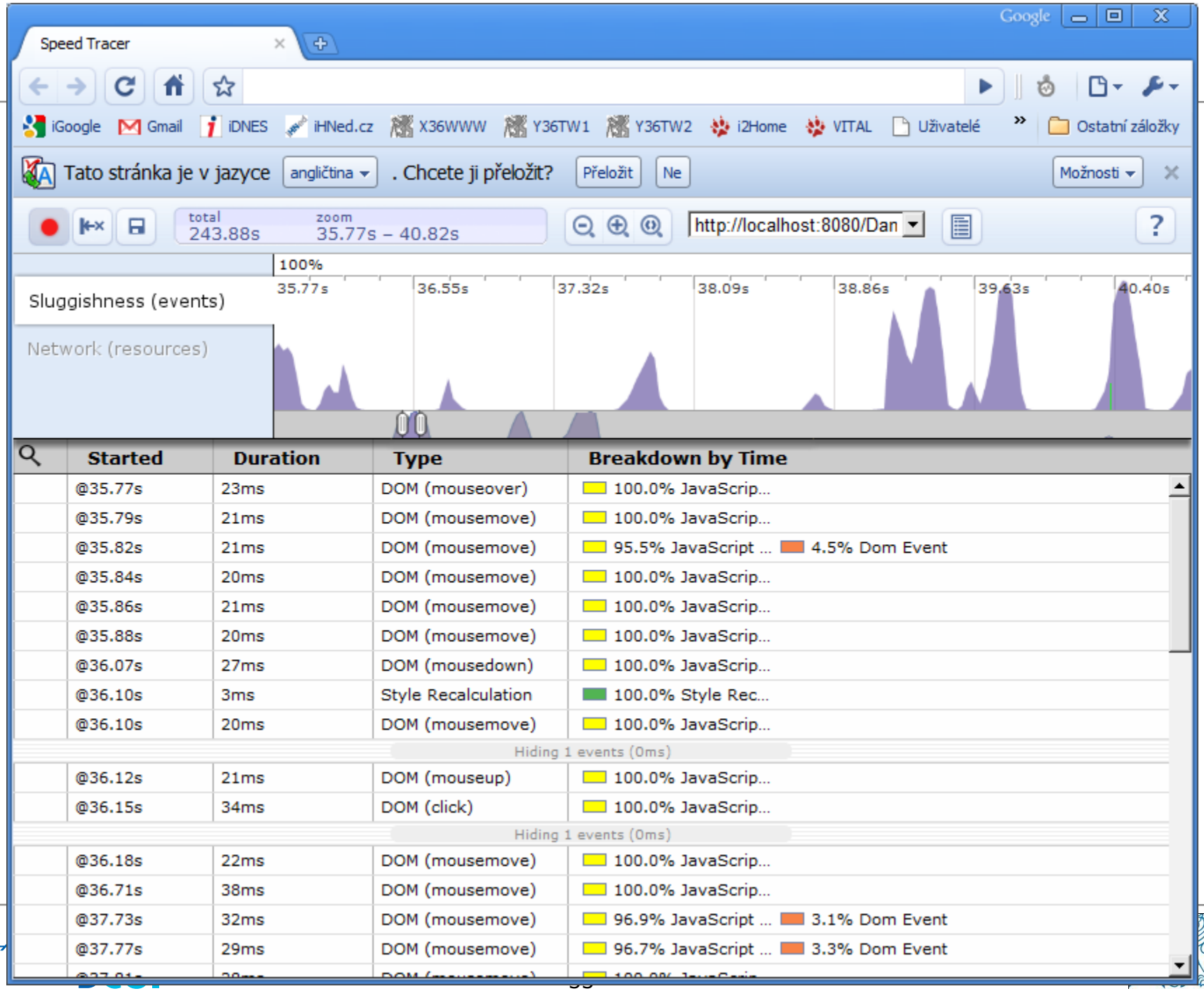

GWT Development Mode
Development Mode
(FF)
Launch GWT Module
Startup URL:
http://localhost:8080/Dama2/
Launch Default Browser
Copy to Clipboard
Expand All
Collapse All

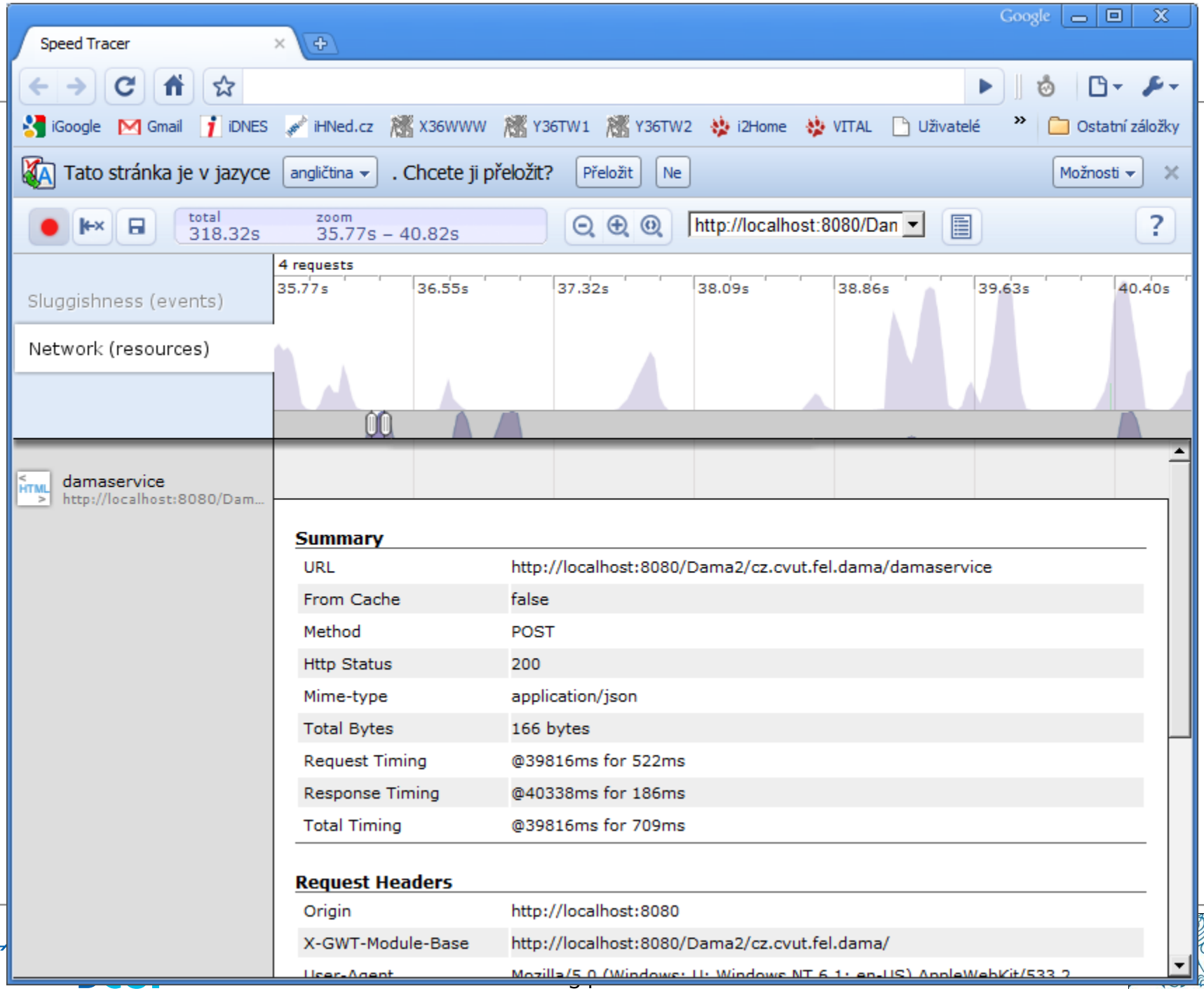
Speed Tracer

- Zajímavý nástroj pro ladění výkonu
- Pro Chrome
- FF má Firebug, který umí něco podobného

- Instalovat Chrome, instalovat plugin pro GWT, SpeedTracer
- Spustit Chrome s přepínačem

  --enable-extension-timeline-api


Speed Tracer
Google
iGoogle
Gmail
iDNES
iHNed.cz
X36WWW
Y36TW1
Y36TW2
i2Home
VITAL
Uživatelé
Ostatní záložky
Tato stránka je v jazyce
angličtina
. Chcete ji přeložit?
Přeložit
Ne
Možnosti
total
243.88s
zoom
35.77s – 40.82s
http://localhost:8080/Dan
100%
Sluggishness (events)
Network (resources)
35.77s
36.55s
37.32s
38.09s
38.86s
39.63s
40.40s
Started
Duration
Type
Breakdown by Time
@35.77s 23ms DOM (mouseover) 100.0% JavaScrip...
@35.79s 21ms DOM (mousemove) 100.0% JavaScrip...
@35.82s 21ms DOM (mousemove) 95.5% JavaScript ... 4.5% Dom Event
@35.84s 20ms DOM (mousemove) 100.0% JavaScrip...
@35.86s 21ms DOM (mousemove) 100.0% JavaScrip...
@35.88s 20ms DOM (mousemove) 100.0% JavaScrip...
@36.07s 27ms DOM (mousedown) 100.0% JavaScrip...
@36.10s 3ms Style Recalculation 100.0% Style Rec...
@36.10s 20ms DOM (mousemove) 100.0% JavaScrip...
Hiding 1 events (0ms)
@36.12s 21ms DOM (mouseup) 100.0% JavaScrip...
@36.15s 34ms DOM (click) 100.0% JavaScrip...
Hiding 1 events (0ms)
@36.18s 22ms DOM (mousemove) 100.0% JavaScrip...
@36.71s 38ms DOM (mousemove) 100.0% JavaScrip...
@37.73s 32ms DOM (mousemove) 96.9% JavaScript ... 3.1% Dom Event
@37.77s 29ms DOM (mousemove) 96.7% JavaScript ... 3.3% Dom Event


Speed Tracer
Google
iGoogle
Gmail
iDNES
iHNed.cz
X36WWW
Y36TW1
Y36TW2
i2Home
VITAL
Uživatelé
Ostatní záložky
Tato stránka je v jazyce
angličtina
. Chcete ji přeložit?
Přeložit
Ne
Možnosti
total
318.32s
zoom
35.77s – 40.82s
http://localhost:8080/Dan
4 requests
Sluggishness (events)
Network (resources)
35.77s
36.55s
37.32s
38.09s
38.86s
39.63s
40.40s
damaservice
http://localhost:8080/Dam...
Summary
URL
http://localhost:8080/Dama2/cz.cvut.fel.dama/damaservice
From Cache
false
Method
POST
Http Status
200
Mime-type
application/json
Total Bytes
166 bytes
Request Timing
@39816ms for 522ms
Response Timing
@40338ms for 186ms
Total Timing
@39816ms for 709ms
Request Headers
Origin
http://localhost:8080
X-GWT-Module-Base
http://localhost:8080/Dama2/cz.cvut.fel.dama/