Google Web Toolkit

Michał Węgorek

ZPO 2009

Plan prezentacji

- Czym jest GWT?
- Co daje GWT – motywacja
- Po co tłumaczyć Javę do JavaScriptu?

  - AJAX niebezpieczeństwa
  - Przewaga GWT nad AJAX
  - RPC
  - Utrzymywanie historii
  - I18N
  - Integracja GWT z istniejącymi aplikacjami
- Omówienie prostej aplikacji GWT

Czym jest GWT?

- Czym jest GWT?
  - zestaw narzędzi do tworzenia aplikacji typu AJAX w Javie
- Dlaczego GWT jest warte uwagi?
  - pisanie, uruchamianie, testowanie i debugowanie wszystkiego w Javie, zarówno kodu po stronie klienta (UI) jak i logiki biznesowej po stronie serwera

Czym jest GWT?

- Gdzieś o tym słyszałem, czy to nie jest aplet?
  -Nie, nie trzeba JVM. GWT przerabia kod w Javie na równoważny kod w czystym JavaScript
- Czyli GWT to kross kompilator?
  Java -> JavaScript?
  -Między innymi, ale to tylko mała część możliwości GWT

Czemu powstało GWT i skąd pomysł na przerabianie Javy do JavaScript?-Motywacja

- Wyższość aplikacji webowych nad aplikacjami innego typu:

  - Każda aplikacja webowa to URL, nie ma instalacji, nie ma brakujących DLLi

  - Bezpieczeństwo:

    Instalowanie = może być niebezpieczne

    Surfowanie po sieci = zwykle bezpieczne

  - Prostota

    Strony mają prosty i przyjemny wygląd

    Nie dużo do nauczenia: wstecz, dalej, guziki, odsyłacze...

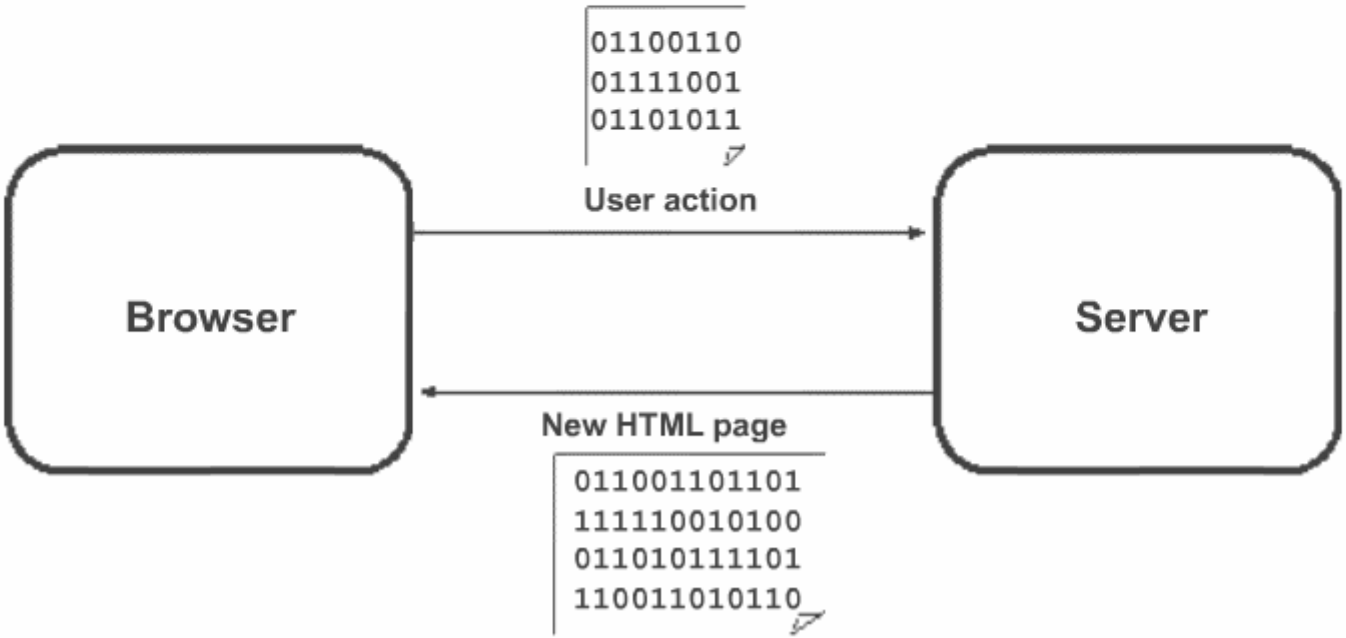
Czemu powstało GWT i skąd pomysł na przerabianie Javy do JavaScript?

Advantage: The architecture is simple

UI is always stateless HTML

Server handles everything

Browsers are HTML dumb terminals



http://www.it-bildungsnetz.de/fileadmin/media/Akademietag_2007/

Czemu powstało GWT i skąd pomysł na przerabianie Javy do JavaScript?

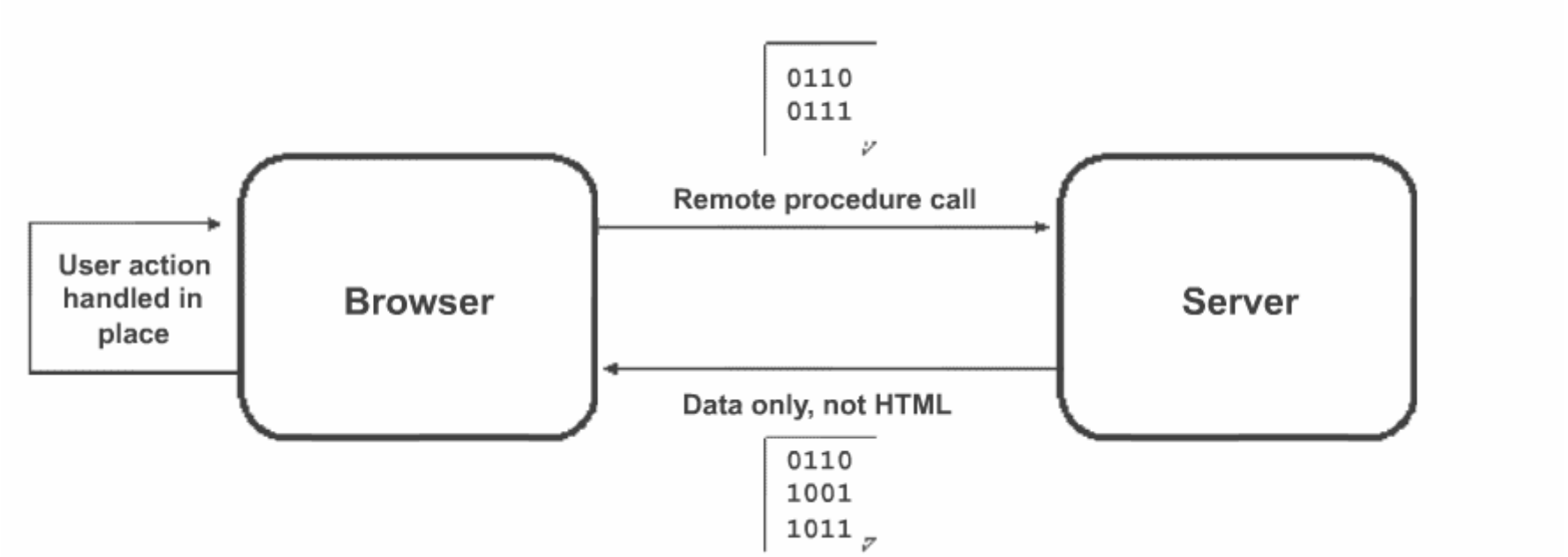
- Google wprowadziło GWT częściowo na swoje potrzeby, m.in.:

  -Gmail, Google Maps...

- problem ze stanem:
  -pojęcie stanu jest sztucznie wprowadzane, bo zapamiętywane po stronie serwera w imieniu klienta (niepotrzebne obciążenie serwera)
- UI nie może się zmieniać interaktywnie u klienta
- aplikacje po stronie klienta mogą być powolne i mogą wymagać wysokiej przepustowości

Use Case

- [przykład – DynaTable]



http://www.it-bildungsnetz.de/fileadmin/media/Akademietag_2007/

Po co tłumaczyć Jave → JavaScript?

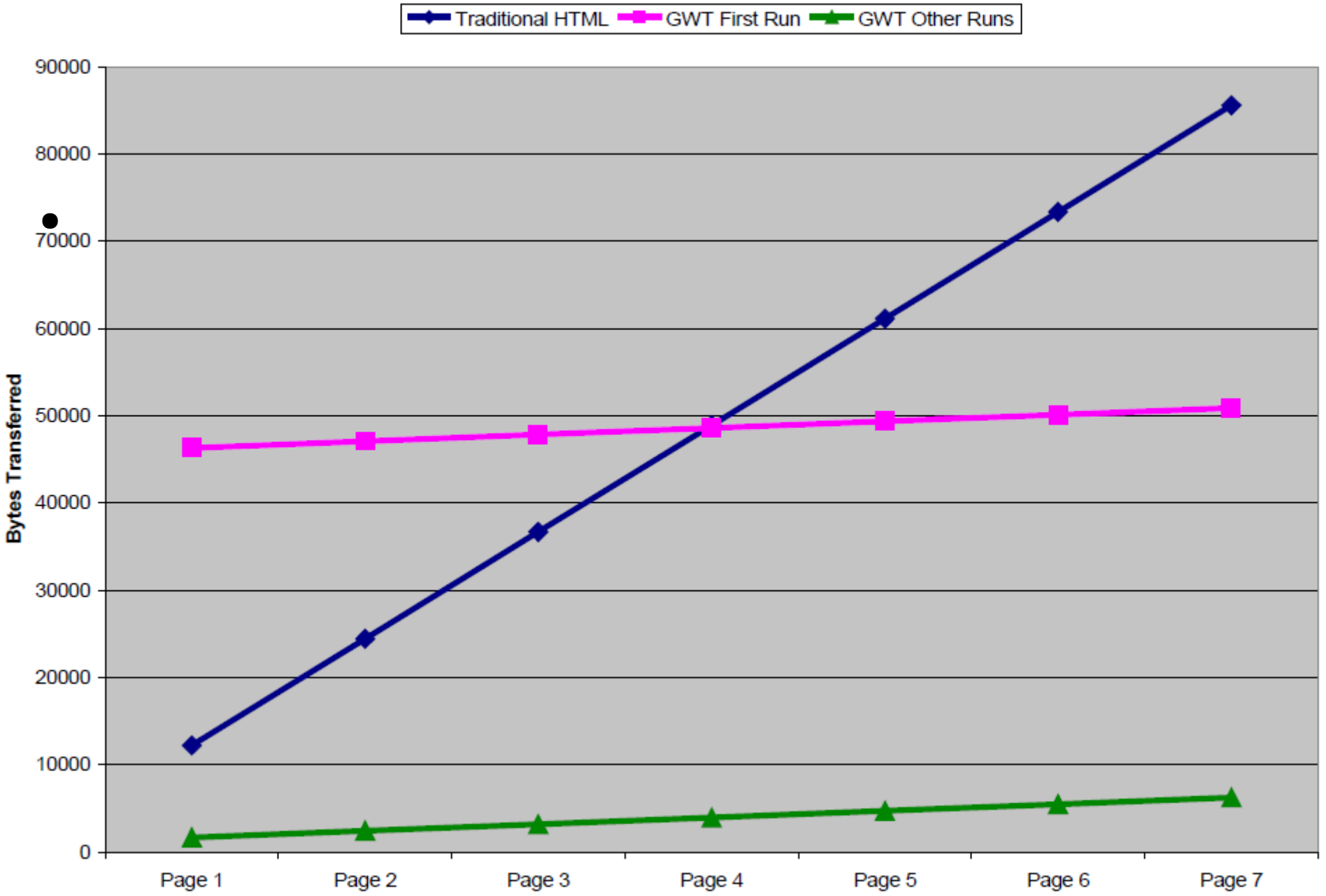
- AJAX niebiezpieczeństwo:
  -trzeba dodać do strony interaktywny skrypt
  -po kilku dniach dowiadujemy się o problemach
  -próbujemy poprawić
  -różnice pomiędzy przeglądarkami zbyt duże
  -nasz kod idzie do kosza

Przewaga GWT nad AJAX

- wykorzystanie Javy, developerów Javy i technologii z nią związanych, dużo potężniejsze narzędzie od AJAX
- debugowanie, JUnit, profiling (dostosowanie kodu do konkretnej przeglądarki)
- zmniejszenie, a nawet wyeliminowanie niezgodności przeglądarek, w GWT nasz kod działa na wszystkich liczących się przeglądarkach (IE, Firefox, Mozilla, Safari, Opera, Chrome).

Przewaga GWT nad AJAX

- Ułatwienie budowania kodu gotowego do ponownego użycia (jar)
- Możliwość zdalnego wywoływania kodu (RPC), bardzo proste przy pomocy GWT
- Przeniesienie (częściowej) informacji o stanie do klienta - odciążenie serwera
- Skalowalność



http://www.it-bildungsnetz.de/fileadmin/media/Akademietag_2007/

Use Case

- [przykład – GWT mail]

RPC

- JSON, XML-RPC
- Idea: chcemy wymieniać się z serwerem obiektami
- GWT RPC, Use Case

  entry-point - punkt dostępowy - klasa realizująca interfejs EntryPoint, która docelowo stanie się stroną HTML. Jest to część kliencka GWT. Może istnieć wiele punktów dostępowych.

  servlet - klasa rozszerzająca klasę RemoteServiceServlet, która jest definicją servletu - części aplikacji wykonywanej po stronie serwera (w sensie GWT i Java EE) i wywoływanej przez mechanizm GWT RPC. Klasa, z której dziedziczy servlet jest jedynie klasą pochodną znanej z Java EE klasy javax.servlet.http.HttpServlet i obsługuje mechanizm serializacji.

GWT RPC – zdalny serwis

- definiujemy interfejs i implementujemy go na serwerze

```
interface SpellService extends RemoteService {
    String[] suggest(String word);
}
```

- Tworzymy proxy i wywołujemy zdalne metody niemalże tak, jakby było one lokalne

```
SpellServiceAsync spell =GWT.create(SpellService.class);
spell.suggest("component", new AsyncCallback() {
    void onSuccess(Object result) {
        ...
    }
    void onFailure(Throwable e) {
        …
    }

};
```

Use Case

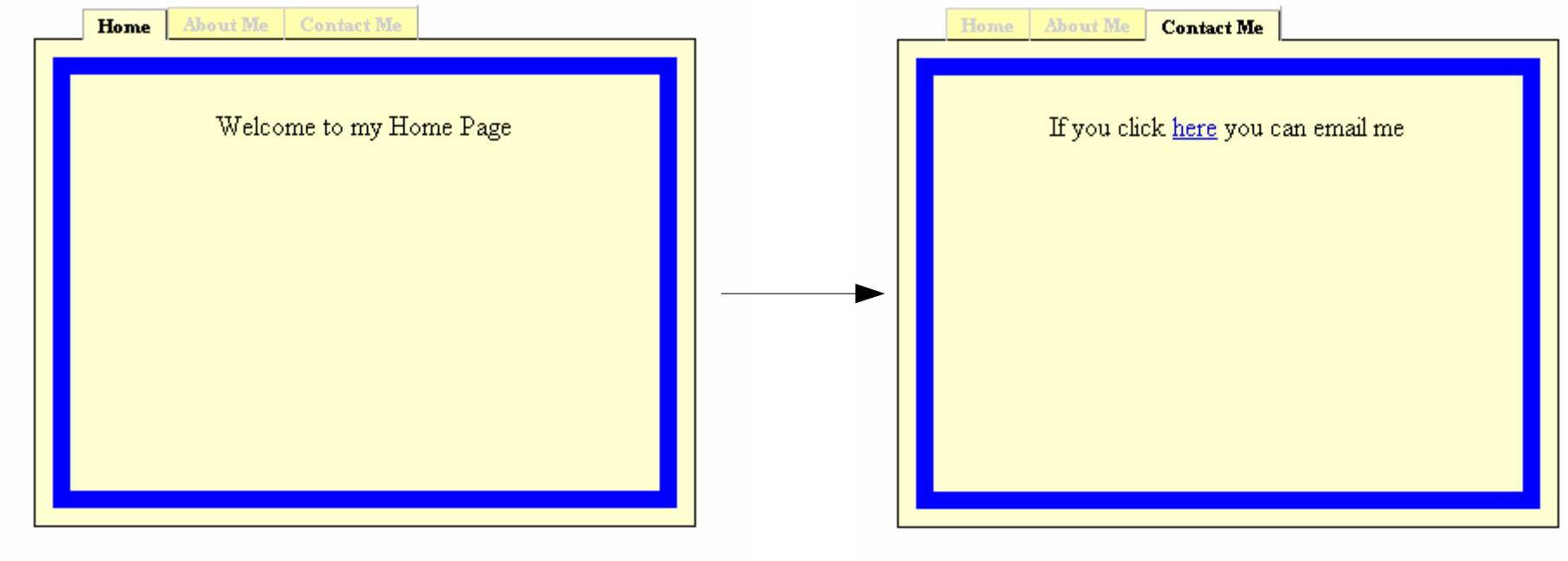
- GWT RPC - WitajŚwiecieGWT

Utrzymywanie historii

- Historia w AJAX
- Historia w GWT

```
public class Main implements EntryPoint, HistoryListener
{
    public void onModuleLoad()
    {
     …
     History.addHistoryListener(this);
     …
     }
    public void onHistoryChanged(String historyToken)
    {...}
}
```

Historia - przykład




```
public void onTabSelected(SourcesTabEvents sender, int tabIndex)
{
    History.newItem(""+tabIndex);
}
```

http://examples.roughian.com/index.htm#Tutorials~History_Support

Historia - przykład

```
String oldToken = null;
public void onHistoryChanged(String historyToken)
{
    // If they are the same, no need to do anything
    if (historyToken != null && historyToken.equals(oldToken)) return;

    // Save the token for the next time round
    oldToken = historyToken;

    ...

    menu.selectTab(index);
}
```

I18N

- INTERNATIONALIZATION = I + 18liter + N
- Jak dodać internacjonalizację do aplikacji

  W MyModule.gwt.xml zaznaczyć, że korzystamy z I18N:

```
<module>
    <inherits name="com.google.gwt.i18n.I18N"/>
</module>
<extend-property name="locale" values="pl,en"/>
```

I18N

Stworzenie plików zawierających teksty stałe dla poszczególnych języków oraz plik domyślny:

Domyślny - MyConstans.properties (w pakiecie com.<..>.client)

helloWorld = Witaj Świecie
goodbyeWorld = Do widzenia Świecie

Polski - MyConstans_pl.properties (w pakiecie com.<..>.client)

helloWorld = Witaj Świecie
goodbyeWorld = Do widzenia Świecie

Angielski MyConstans_en.properties (w pakiecie com.<..>.client)

helloWorld = Hello World
goodbyeWorld = Goodbye World

I18N

- stworzenie w pakiecie com.<..>.client interfejsu MyConstans, w którym nazwy metod będą odpowiadać kluczom w plikach z tekstami stałymi:

```
public interface MyConstants extends Constants {
  String helloWorld();
  String goodbyeWorld();
}
```

I18N

- Aby użyc naszych tekstów, wystarczy:

  1) MyConstants myConstants = (MyConstants)GWT.create(MyConstants.class);

  2) Od tej pory można używac metod obiektu myConstants:

  Window.alert(myConstants.helloWorld());

  Biblioteka I18N sama troszczy się o język. Sprawdza atrybut locale podany w URL'u. Np: http://www.mycompany.com/MyModule.html?locale=pl

Integracja modułu GWT

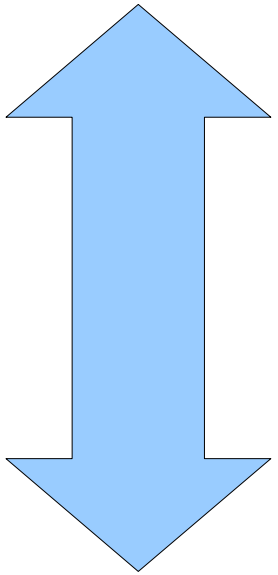
```
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>
<html>
  <head>
    <title>
      JSP 2.0 Examples -
      Display Products Tag File
    </title>
    <link
      type="text/css"
      rel="stylesheet"
      href="script/DynaTable.css"/>
    <meta
      name='gwt:module'
      content=
        'script=com.google.gwt.sample.dynatable.DynaTable'>
    <script src='script/gwt.js'></script>
  </head>
  <body>
    <h1>JSP 2.0 Examples - Display Products Tag File</h1>
...
```

http://www.it-bildungsnetz.de/fileadmin/media/Akademietag_2007/

Integracja modułu GWT

- W kodzie Java, w metodzie onModuleLoad() klienta:

```
RootPanel.get("stockList").add(dowolnyWidgetGWT);
```



- W kodzie HTML:

```
<div id="stockList"></div>
```

Omówienie prostej aplikacji

- StockWatcher http://code.google.com/intl/pl/webtoolkit/tutorials/1.6/create.html

Bibliografia

- http://www.it-bildungsnetz.de/fileadmin/media/Akademietag_2007/ - grafika
- http://www.infoq.com/ - using google web toolkit
- http://coderlife.blogspot.com/2008/04/gwt-co-nacja-to-inna.html - internacjonalizacja
- http://www.jaceklaskowski.pl/blog/2007/05/02/gwt-rpc-mechanizm-zdalnego-
- http://code.google.com/intl/pl/webtoolkit/
- http://examples.roughian.com/index.htm#Tutorials~History_Support