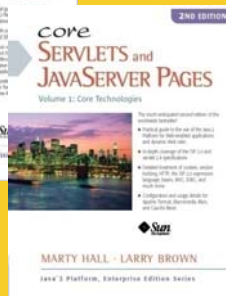
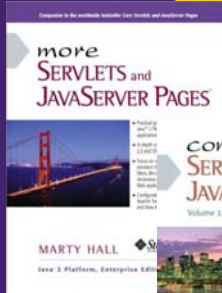




# The Google Web Toolkit (GWT): Widget Event Handling (GWT 2.5 Version)

Originals of Slides and Source Code for Examples:  
<http://courses.coreservlets.com/Course-Materials/gwt.html>

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



**For live Ajax & GWT training, see training courses at <http://courses.coreservlets.com/>.**



**Taught by the author of *Core Servlets and JSP*, *More Servlets and JSP*, and this tutorial. Available at public venues, or customized versions can be held on-site at your organization.**

- Courses developed and taught by Marty Hall
  - JSF 2, PrimeFaces, servlets/JSP, Ajax, jQuery, Android development, Java 7 or 8 programming, custom mix of topics
  - Courses available in any state or country. Maryland/DC area companies can also choose afternoon/evening courses.
- Courses developed and taught by coreservlets.com experts (edited by Marty)
  - Spring, Hibernate/JPA, GWT, Hadoop, HTML5, RESTful Web Services

Contact [hall@coreservlets.com](mailto:hall@coreservlets.com) for details



# Topics in This Section

- **Main approaches to event handling**
  - Separate event handler classes
  - Main class implementing event handler interface
  - Named inner classes
  - Anonymous inner classes
- **Basic widgets and their associated events**
  - Pushbuttons and related widgets
  - Checkboxes and related widgets
  - Listboxes and related widgets
  - Textfields and related widgets

5

© 2013 Marty Hall & Yaakov Chaikin



## Idea

**Customized Java EE Training: <http://courses.coreservlets.com/>**

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# General Approach

- **Make a control**
  - `TextBox textfield = new TextBox();`
- **Attach an event handler (4 alternative ways)**
  - `textfield.addKeyUpHandler(new MyKeyUpHandler());`
    - `MyKeyUpHandler` is class that implements `KeyUpHandler`
  - `textfield.addKeyUpHandler(this);`
    - Current class (`this`) implements `KeyUpHandler`
  - `textfield.addKeyUpHandler (new MyKeyUpHandler());`
    - `MyKeyUpHandler` is inner class that implements `KeyUpHandler`
  - `textfield.addKeyUpHandler(new KeyUpHandler () { ... });`
- **Note**
  - This is *exactly* same approach as with Swing, AWT, SWT, Android
    - You write event handlers using the same strategies as in desktop Java GUI programming, even though code gets compiled to JavaScript and HTML at runtime
      - Client-side code will not support Java 8 for foreseeable future, so you cannot use lambdas for event handling. Too bad!

7

© 2013 Marty Hall & Yaakov Chaikin



## Option 1: Separate Handler Classes

Customized Java EE Training: <http://courses.coreservlets.com/>

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Idea

- **Monitor a textfield**
  - Echo contents of textfield to table cell
  - If textfield contents match “gwt”, give special message
- **Approach**
  - Use an external class that implements KeyUpHandler
- **Advantages**
  - Separate classes generally promote loose coupling
    - Event handler can be changed independently from rest of app
  - Works even if you have multiple textfields to which you attach different handlers
  - You can pass arguments to the constructor to customize behavior
- **Disadvantages**
  - If you want to call code in main class, you need reference
  - Even then, code in main class must be public

9

# Main Class (Initial Attempt)

```
public class GwtEvents1 implements EntryPoint {
    private TextBox textfield;
    public HTML resultArea;

    public void onModuleLoad() {
        textfield = new TextBox();
        textfield.addKeyUpHandler(new WowHandler(this));
        resultArea = new HTML("<i>Result will go here</i>");
        RootPanel.get().addStyleName("tan");
        RootPanel.get("textfieldID").add(textfield);
        RootPanel.get("resultID").add(resultArea);
    }

    public void backgroundRed() {
        RootPanel.get().addStyleName("red");
    }

    public void backgroundNormal() {
        RootPanel.get().removeStyleName("red");
    }
}
```

10

# Handler Class (Initial Attempt)

```
public class WowHandler implements KeyUpHandler {
    private GwtEvents1 app;

    public WowHandler(GwtEvents1 app) {
        this.app = app;
    }

    public void onKeyUp(KeyUpEvent event) {
        TextBox textfield = (TextBox)event.getSource();
        String text = textfield.getText();
        if(text.equalsIgnoreCase("gwt")) {
            app.resultArea.setHTML("<span class='wow'>Wow!</span>");
            app.backgroundRed();
        } else {
            app.resultArea.setHTML(text);
            app.backgroundNormal();
        }
    }
}
```

You need to call main application, so you need to store and use a reference to it. Data in main app must be public.

11

# Main Class (Improved)

```
public class GwtEvents1 implements EntryPoint, RespondsToWow {
    private TextBox textfield;
    private HTML resultArea;

    public void onModuleLoad() {
        ...
    }
    @Override
    public void makeBackgroundRed() {
        RootPanel.get().addStyleName("red");
    }
    @Override
    public void makeBackgroundNormal() {
        RootPanel.get().removeStyleName("red");
    }
    @Override
    public void setResults(String html) {
        resultArea.setHTML(html);
    }
}
```

12



# Interface – RespondsToWow

```
package coreservlets.client;

import com.google.gwt.user.client.ui.HTML;

public interface RespondsToWow {
    void setResults(String html);

    void makeBackgroundRed();

    void makeBackgroundNormal();
}
```


13

# Handler Class (Improved)

```
public class WowHandler implements KeyUpHandler {
    private RespondsToWow app;

    public WowHandler(RespondsToWow app) {
        this.app = app;
    }

    public void onKeyUp(KeyUpEvent event) {
        TextBox textfield = (TextBox) event.getSource();
        String text = textfield.getText();
        if (text.equalsIgnoreCase("gwt")) {
            app.setResults("<span class='wow'>Wow!</span>");
            app.makeBackgroundRed();
        } else {
            app.setResults(text);
            app.makeBackgroundNormal();
        }
    }
}
```

 No longer tied to concrete class!

14

# HTML

```
...
<link rel="stylesheet"
      href="./css/styles.css"
      type="text/css"/>
<script type="text/javascript" language="javascript"
        src="gwtevents1/gwtevents1.nocache.js"></script>
</head>
<body>
...
<table border="1">
  <tr><th>Input</th>
    <th>Result</th></tr>
  <tr><td id="textfieldID"></td>
    <td id="resultID"></td></tr>
</table><br/>
...

```

Left in from auto-generated code

ids match values passed to RootPanel.get in main Java class

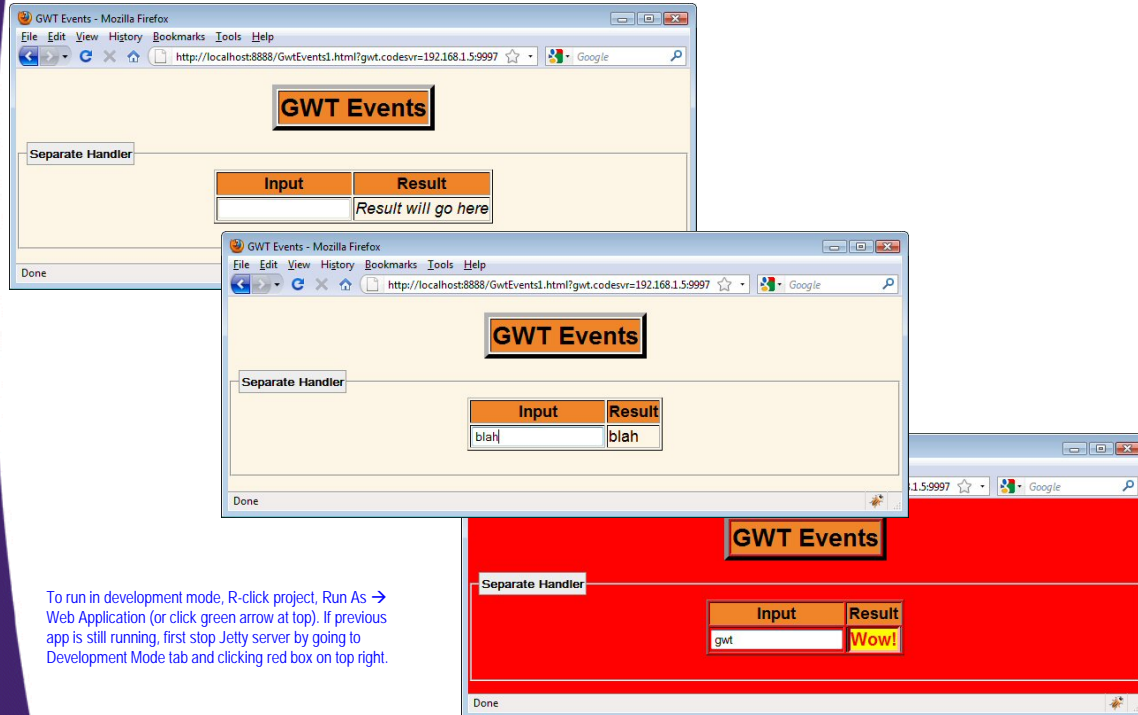
15

# CSS (Same in All Examples)

```
.red { background-color: red }
.wow { background-color: yellow;
      color: red;
      font-weight: bold;
      font-size: 20px;
      font-family: Arial, Helvetica, sans-serif;
      border-style: inset;
}
...
```

16

# Results (Development Mode)



To run in development mode, R-click project, Run As → Web Application (or click green arrow at top). If previous app is still running, first stop Jetty server by going to Development Mode tab and clicking red box on top right.

17

© 2013 Marty Hall & Yaakov Chaikin



## Option 2: Main Class Implementing Handler Interface

Customized Java EE Training: <http://courses.coreservlets.com/>

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android. Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



# Idea

- **Monitor a textfield**
  - Echo contents of textfield to table cell
  - If textfield contents match “gwt”, give special message
- **Approach**
  - Have main class implement KeyUpHandler
- **Advantages**
  - No need to pass main class reference to handler
  - Handler can easily access data in the main class
  - Methods or variables in main class can be private
- **Disadvantages**
  - Extra work if you have multiple textfields to which you attach different behaviors
    - You need if/then/else in the event handler to determine which textfield you are dealing with

19

# Main Class

```
public class GwtEvents2 implements EntryPoint, KeyUpHandler {
    private TextBox textfield;
    private HTML resultArea;

    public void onModuleLoad() {
        textfield = new TextBox();
        textfield.addKeyUpHandler(this);
        resultArea = new HTML("<i>Result will go here</i>");
        RootPanel.get().addStyleName("tan");
        RootPanel.get("textfieldID").add(textfield);
        RootPanel.get("resultID").add(resultArea);
    }

    private void makeBackgroundRed() {
        RootPanel.get().addStyleName("red");
    }

    private void makeBackgroundNormal() {
        RootPanel.get().removeStyleName("red");
    }
}
```

20

## Main Class (Continued)

```
public void onKeyUp(KeyUpEvent event) {
    String text = textfield.getText();
    if(text.equalsIgnoreCase("gwt")) {
        resultArea.setHTML("<span class='wow'>Wow!</span>");
        makeBackgroundRed();
    } else {
        resultArea.setHTML(text);
        makeBackgroundNormal();
    }
}
}
```

Can easily call methods in main class (even private ones), since onKeyUp is part of main class.

21

## HTML (Same As Before Except for JavaScript File Name)

```
...
<link rel="stylesheet"
      href="./css/styles.css"
      type="text/css"/>
<script type="text/javascript" language="javascript"
        src="gwtevents2/gwtevents2.nocache.js"></script>
</head>
<body>
...
<table border="1">
  <tr><th>Input</th>
    <th>Result</th></tr>
  <tr><td id="textfieldID"></td>
    <td id="resultID"></td></tr>
</table><br/>
...

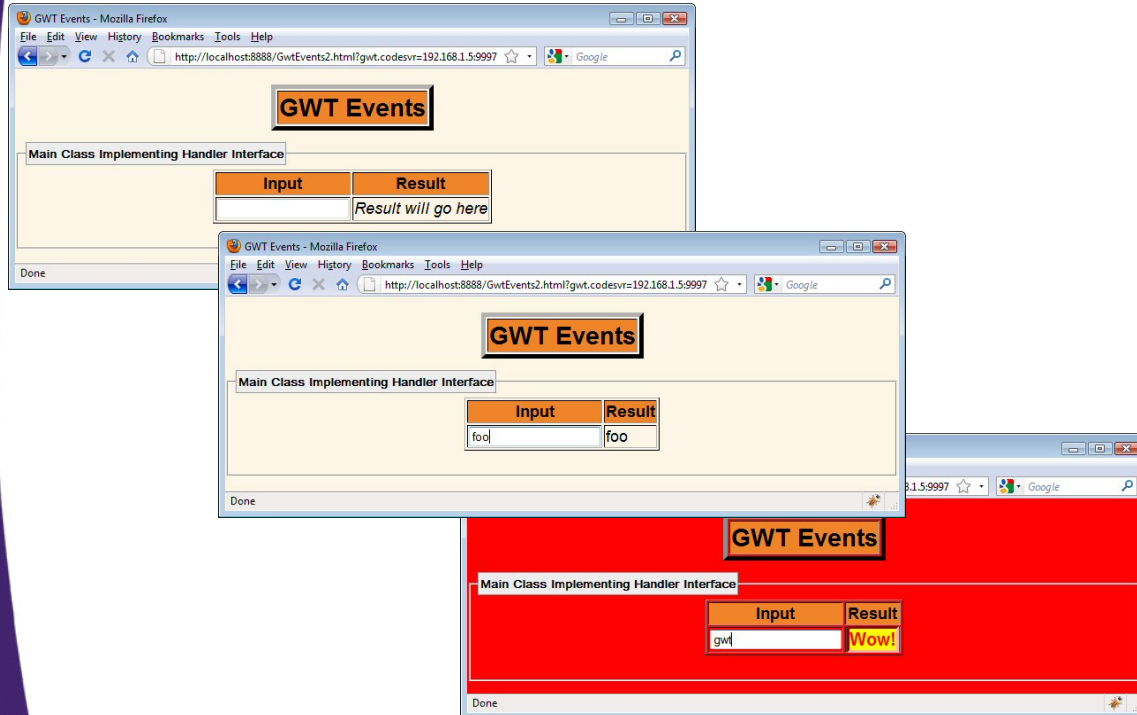
```

Left in from auto-generated code

ids match values passed to RootPanel.get in main Java class

22

# Results (Development Mode)



23

© 2013 Marty Hall & Yaakov Chaikin



## Option 3: Named Inner Classes

Customized Java EE Training: <http://courses.coreservlets.com/>

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Idea

- **Monitor a textfield**
  - Echo contents of textfield to table cell
  - If textfield contents match “gwt”, give special message
- **Approach**
  - Use inner class that extends KeyUpHandler
- **Advantages**
  - Same advantages as interface (access to main class data)
  - Plus, would work for abstract adapter classes
  - Scales well if you have multiple textfields with different behaviors
- **Disadvantages**
  - Little advantage (vs. implementing interface) if handler interface has only one method and you have only one textfield
    - Most GWT handlers have only one method
  - Slightly more complex than interface approach

25

# Main Class

```
public class GwtEvents3 implements EntryPoint {
    private TextBox textfield;
    private HTML resultArea;

    public void onModuleLoad() {
        textfield = new TextBox();
        textfield.addKeyUpHandler(new WowHandler());
        resultArea = new HTML("<i>Result will go here</i>");
        RootPanel.get().addStyleName("tan");
        RootPanel.get("textfieldID").add(textfield);
        RootPanel.get("resultID").add(resultArea);
    }

    private void makeBackgroundRed() {
        RootPanel.get().addStyleName("red");
    }

    private void makeBackgroundNormal() {
        RootPanel.get().removeStyleName("red");
    }
}
```

26

## Main Class (Continued)

```
private class WowHandler implements KeyUpHandler {
    public void onKeyUp(KeyUpEvent event) {
        String text = textfield.getText();
        if(text.equalsIgnoreCase("gwt")) {
            resultArea.setHTML("<span class='wow'>Wow!</span>");
            makeBackgroundRed();
        } else {
            resultArea.setHTML(text);
            makeBackgroundNormal();
        }
    }
}
```

27

## HTML (Same As Before Except for JavaScript File Name)

```
...
<link rel="stylesheet"
      href="./css/styles.css"
      type="text/css"/>
<script language='javascript'
        src='coreservlets.GwtEventsApp3.nocache.js'>
</script>
</head>
<body>
...
<table border="1">
  <tr><th>Input</th>
    <th>Result</th></tr>
  <tr><td id="textfieldID"></td>
    <td id="resultID"></td></tr>
</table><br/>
...

```

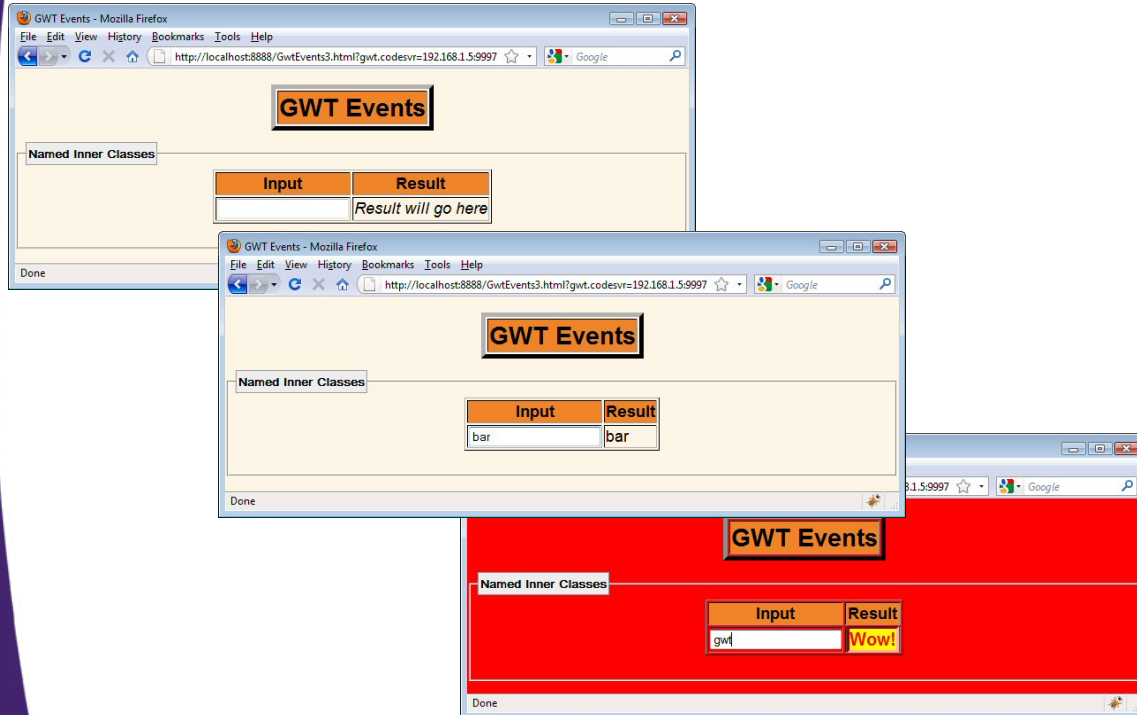
Left in from auto-generated code

ids match values passed to RootPanel.get in main Java class

28



# Results (Development Mode)



29

© 2013 Marty Hall & Yaakov Chaikin



## Option 4: Anonymous Inner Classes

Customized Java EE Training: <http://courses.coreservlets.com/>

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android. Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

## Idea

- **Monitor a textfield**
  - Echo contents of textfield to table cell
  - If textfield contents match “gwt”, give special message
- **Approach**
  - Use anonymous inner class that extends KeyUpHandler
- **Advantages**
  - Same advantages as inner classes
  - Shorter
- **Disadvantages**
  - Handler cannot be reused for multiple controls
  - May have small performance penalties in resultant JavaScript if there are many anonymous inner classes

31

## Main Class

```
public class GwtEvents4 implements EntryPoint {
    private TextBox textfield;
    private HTML resultArea;

    private void makeBackgroundRed() {
        RootPanel.get().addStyleName("red");
    }

    private void makeBackgroundNormal() {
        RootPanel.get().removeStyleName("red");
    }
}
```

32

# Main Class (Continued)

```
public void onModuleLoad() {
    textfield = new TextBox();
    textfield.addKeyUpHandler(new KeyUpHandler() {
        public void onKeyUp(KeyUpEvent event) {
            String text = textfield.getText();
            if(text.equalsIgnoreCase("gwt")) {
                resultArea.setHTML("<span class='wow'>Wow!</span>");
                makeBackgroundRed();
            } else {
                resultArea.setHTML(text);
                makeBackgroundNormal();
            }
        }
    });
    resultArea = new HTML("<i>Result will go here</i>");
    RootPanel.get("textfieldID").add(textfield);
    RootPanel.get("resultID").add(resultArea);
}
```

Inner-class code refers only to instance variables and new local variables. If you refer to local variables of the outside code, those variables must be declared final. This is not particular to GWT, but is a general requirement in Java for anonymous inner classes.

This is also a perfect place for Java 8 lambdas, but sadly, the client-side GWT code will not support Java 8 for foreseeable future.

33

# HTML (Same As Before Except for JavaScript File Name)

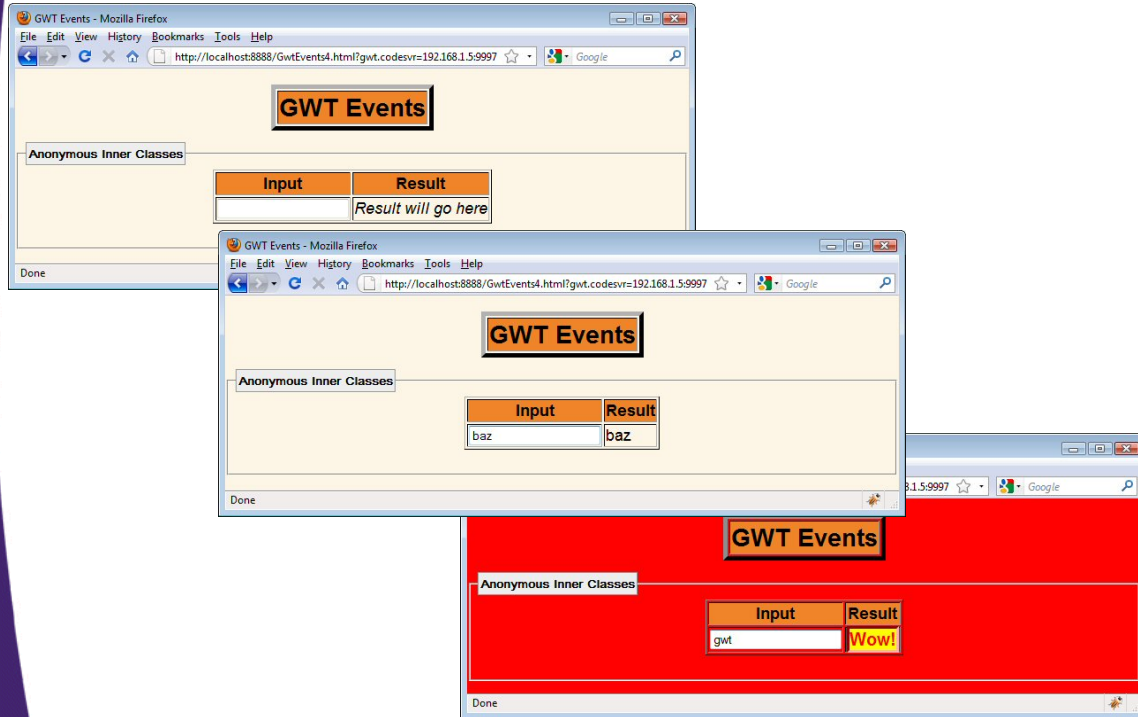
```
...
<link rel="stylesheet"
      href="./css/styles.css"
      type="text/css"/>
<script type="text/javascript" language="javascript"
        src="gwtevents4/gwtevents4.nocache.js"></script>
</head>
<body>
...
<table border="1">
  <tr><th>Input</th>
    <th>Result</th></tr>
  <tr><td id="textfieldID"></td>
    <td id="resultID"></td></tr>
</table><br/>
...
```

Left in from auto-generated code

ids match values passed to RootPanel.get in main Java class

34

# Results (Development Mode)



35

© 2013 Marty Hall & Yaakov Chaikin



## Basic Widgets and Their Handlers

Customized Java EE Training: <http://courses.coreservlets.com/>

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android. Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Compatibility Note

- **Early GWT versions**
  - Used *BlahListener* (with *Widget* as argument)
    - *ClickListener*, *FocusListener*, etc.
  - Had a single keyboard listener
    - *KeyboardListener*
- **Recent GWT versions (GWT 1.5+)**
  - Use *BlahHandler* (with *BlahEvent* as argument)
    - *ClickHandler*, *FocusHandler*, etc.
  - Have multiple keyboard listeners
    - *KeyUpHandler*, *KeyDownHandler*, *KeyPressHandler*
- **GXT (from extjs.com)**
  - Called *BlahListener*, but more closely follows the GWT 1.5 approach

37

# Button and PushButton

- **Purpose**
  - To initiate an action
    - *Button* is normal HTML button
    - *PushButton* is custom image button
- **Main handler**
  - *ClickHandler*
    - *onClick(ClickEvent event)*
- **Auxiliary handlers**
  - *FocusHandler*
    - *onFocus(Widget sender)*
  - *KeyDownHandler*, *KeyUpHandler*, *KeyPressHandler*
    - *onKeyDown*, *onKeyUp*, *onKeyPress* (see *TextBox*)



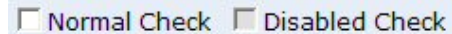
38



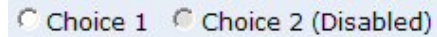
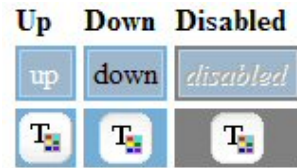
# CheckBox, ToggleButton, and RadioButton

- **Purpose**

- To choose between two options
  - CheckBox, ToggleButton
- To choose among several options
  - RadioButton



Normal Check    Disabled Check



Choice 1    Choice 2 (Disabled)

- **Main handler**

- ClickHandler
  - `onClick(ClickEvent event)`

- **Auxiliary handlers**

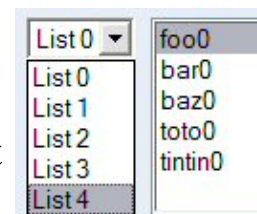
- FocusHandler
  - `onFocus(FocusEvent event)`
- KeyDownHandler, KeyUpHandler, KeyPressHandler
  - `onKeyDown`, `onKeyUp`, `onKeyPress` (see TextBox)

39

# ListBox

- **Purpose**

- To choose one of options from list
  - `new ListBox(false)`
- To choose several of options from list
  - `new ListBox(true)`



- **Main handler**

- ChangeHandler
  - `onChange(ChangeEvent event)`

- **Auxiliary handlers**

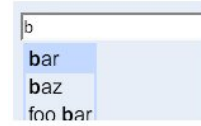
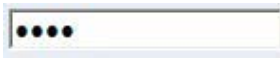
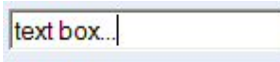
- ClickHandler
- FocusHandler
- KeyDownHandler, KeyUpHandler, KeyPressHandler

40

# TextBox, PasswordTextBox, TextArea, SuggestBox

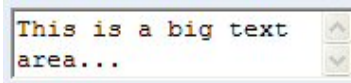
- **Purpose**

- To collect text



- **Main handlers**

- KeyDownHandler, **KeyUpHandler**, KeyPressHandler
  - onKeyDown(KeyDownEvent event)
  - **onKeyUp(KeyUpEvent event)**
  - onKeyPress(KeyPressEvent event)



- **Auxiliary handlers**

- ChangeHandler
- ClickHandler
- FocusHandler

41

© 2013 Marty Hall & Yaakov Chaikin



## Wrap-Up

**Customized Java EE Training: <http://courses.coreservlets.com/>**

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Summary

- **Separate handler classes**
  - Pro: more modular (use interfaces to decouple)
  - Cons: hard to call code in main app
  - Least common approach
- **Main class implementing handler interface**
  - Pro: easy to call code in main app
  - Cons: harder if multiple methods in interface or multiple controls
  - Easiest when interface has one method and I am handling only one control with that event type
- **Named inner classes**
  - Pro: good if you have multiple textfields with different behaviors
  - Cons: slightly more complex
  - My favorite when interface has multiple methods or when handling multiple controls with the same event types. My overall favorite.
- **Anonymous inner classes**
  - Pro: short and succinct
  - Cons: more complex, small performance penalty in some cases
  - Favorite approach of many Swing, SWT, and Android developers

43

© 2013 Marty Hall & Yaakov Chaikin



## Questions?

[JSF 2](#), [PrimeFaces](#), [Java 7 or 8](#), [Ajax](#), [jQuery](#), [Hadoop](#), [RESTful Web Services](#), [Android](#), [HTML5](#), [Spring](#), [Hibernate](#), [Servlets](#), [JSP](#), [GWT](#), and other [Java EE training](#).

**Customized Java EE Training: <http://courses.coreservlets.com/>**

GWT, Java 7 and 8, JSF 2, PrimeFaces, HTML5, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, REST, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.