jQueryUI Autocomplete

Widget of the Week (WoW) Series

Administrative
Computing &
Telecommunications

UC San Diego

Things to know …

- jQuery
  - Third-party JavaScript library
- jQueryUI
  - “Official” widget library built on top of jQuery
- JSON
  - JavaScript Object Notation
- AJAX
  - Asynchronous JavaScript and X(HT)ML

jQuery

- Arguably the most popular JavaScript library in use today
- Excels at DOM manipulation, event handling, client-server interaction
- Good cross-browser support
- Readily extensible through plug-ins and libraries …

jQueryUI

- Library of widgets, effects, events
- Extension to core jQuery

JSON

- “JavaScript Object Notation”
- Lightweight data interchange format
- Arguably human-readable
- Based on object and array literal syntax

AJAX

- “Asynchronous JavaScript and X(HT)ML”
- Technique (not a framework) often implemented with jQuery
- Information retrieved from server with JavaScript XMLHttpRequest
- Displayed on page by manipulating DOM/CSS with JavaScript
- Major restriction: “same origin policy”

Use the 5Ps approach:

- Prepare
- Plan
- Prototype
- Plumb
- Proselytize

Use the 5Ps approach:

- Prepare
- Plan
- Prototype
- Plumb
- Proselytize
- Phone Ike

Use the 5Ps approach:

- Prepare
- Plan
- Prototype
- Plumb
- Proselytize
- Secure

Prepare

- Which widget(s) do I want/need to use?
  - jQueryUI Autocomplete
- What docs are available?
  - jQuery / jQueryUI online docs, uxt.ucsd.edu
- What client-side resources are required?
  - jQuery core, jQueryUI, HTML text input field
- What server-side resources are required?
  - Data source and view for autocomplete
  - JSON libraries for SpringMVC

Plan

- Use legacy data source (JLink business object, etc.) with SpringMVC JSON view
- … or … use existing JLink service

Prototype

- Legacy business object will typically retrieve a List of Map<String,String>
- When added to a SpringMVC JSON view, this will be rendered as a JavaScript array of objects …

Prototype (cont'd)

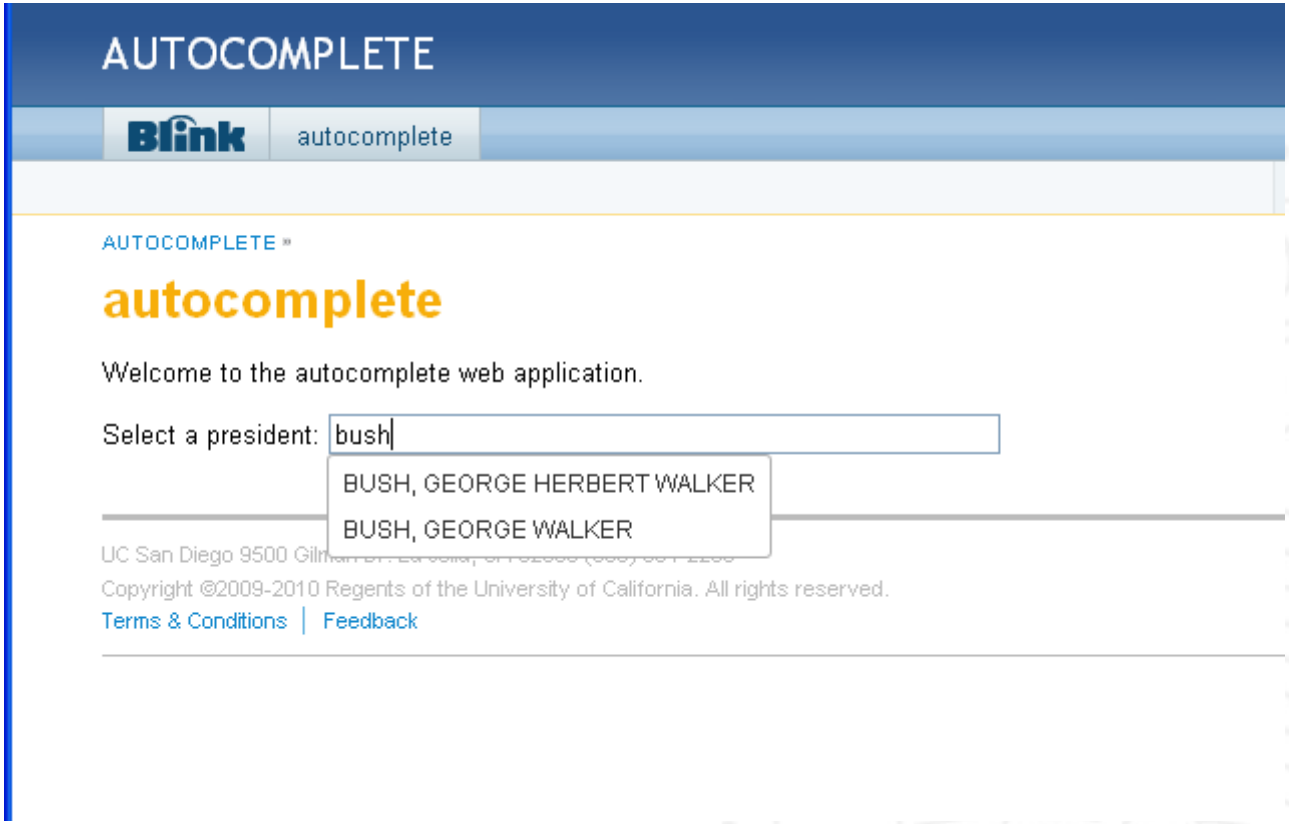

```
{"presidents":
   [
     {"name":"BUSH, GEORGE W.", pob:"NEW HAVEN, CT"},
     {"name":"BUSH, GEORGE H.W.", pob:"MILTON, MA"}
   ]
}
```

Prototype (cont'd)

- Add HTML input field and associated JavaScript
- Make sure you're using jQueryUI >= 1.8.5

```
<form>
    <input type="text" name="president" id="president" size="40" />
</form>
<script type="text/javascript">
    ( function($) {
        $(document).ready( function() {
            $("#president").autocomplete({
                minLength : 3,
                source : function(req, add) {
                    $.getJSON("testData.js", req, function(jsonData) {
                    …
```

Prototype (cont'd)



Plumb

- Define a JSON view in SpringMVC
  - Add spring-json-1.1-act.jar and sojo-optional-0.5.0.jar to webapp
  - Define a JSON view in views.xml
- Create a Spring controller using JSON view
- Test your new JSON view and controller
- Plug your autocomplete widget into your new JSON controller
- See tutorial project for details

Proselytize

- Tell your friends!

Secure

- How do you keep unauthorized users from gaining access to your services?
  - Use Jlink Security Filter to control access by role
  - Roll (role?) your own session-based authentication
  - Use Spring Security (a subject for another lunch hour)

But wait, there's more!

- What about event handlers …

```
select : function(event, ui) {
    console.log(ui);
    alert("You've selected " + ui.item.value);
}
```

But wait, there's still more!

But wait, there's still more!

- What about plumbing this to a JLink REST service?…

```
<form>
    <input type="text" name="president" id="president" size="40" />
</form>
<script type="text/javascript">
    ( function($) {
        $(document).ready( function() {
            $("#president").autocomplete({
                minLength : 3,
                source : function(req, add) {
                    $.getJSON("/jlinkesb/json/demo_affiliate/name/" + req.term,
req, function(jsonData) {
                    …
```

Middleware, UI, and database

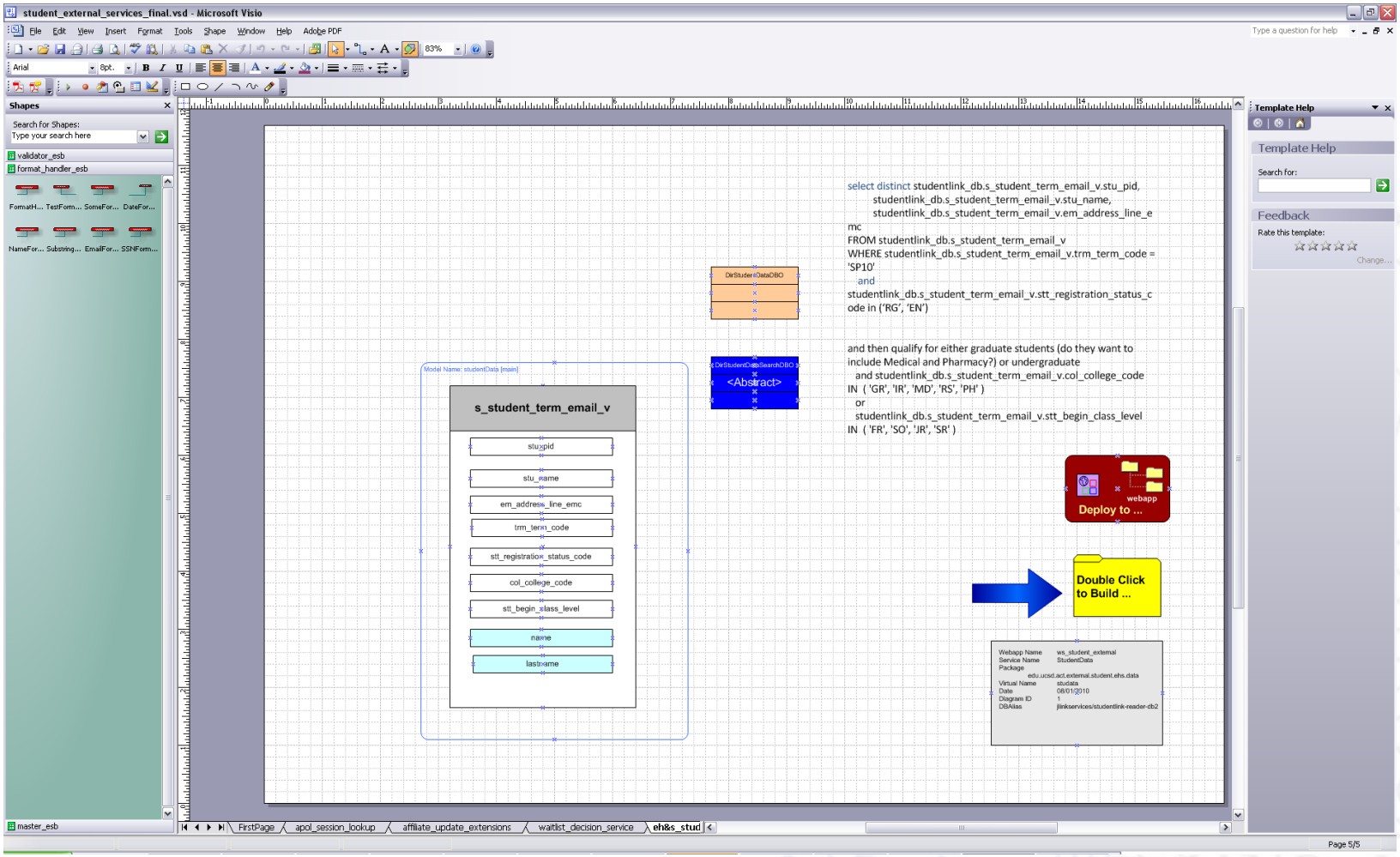
Optional exercise

- ReST Service Adaptor
- Soundex Demo
- Service Diagram
- Service Implementation
- DAO Implementation

ReST Config

- Affiliate Service
- <ServiceDefinition>
- <Service name="jlinkservices:AffiliateLookup" extname="affiliate" />
  <setValue attribute="user.cn" tag="name" /></Service>
- <Cachable/>
- <Attributes nametag = "presidents">
- <Attribute name="user.cn" tag="name" />
- <Attribute name="user.aid" tag="aid" />
- </Attributes>
- </ServiceDefinition>
- URL
- /jlinkesbsso/json/demo_affiliate/name/<in>

Service Diagram



Service Implementation

```
public class DirStudentDataSearchDBO extends DirStudentDataSearchDBOAbstract {

    public DirStudentDataSearchDBO() throws Exception {
        super();
    }

    @Override
    public void assignCaseCriteria(BusinessObject dbo) throws BusinessObjectException {

        dbo.setKey("studata.trm_term_code", true);
        dbo.setDistinct(true);
        String[] l0 = {"RG", "EN"};
        dbo.setCriteria(BusinessObjectCriteria.AND, "studata.stt_registration_status_code", BusinessObjectCriteria.EQUALS, "RG");

    }
    Soundex soundex = new Soundex();
    @Override
    public String getStringValue(String property) {
        if (property.endsWith(".name")) {
            try {
                int idx = super.getStringValue(property).indexOf(",");
                return soundex.soundex(super.getStringValue(property).substring(0, idx)).trim();
            } catch (Exception e) {
                return "";
            }
        }
        if (property.endsWith(".lastname")) {
            try {
                int idx = super.getStringValue(property).indexOf(",");
                return super.getStringValue(property).substring(0, idx).trim();
            } catch (Exception e) {
                return "";
            }


        }                return super.getStringValue(property);

    }
}
```

DAO Implementation

```
public class NameDao {

  private static BusinessObjectPersistenceManager bopm = null;
  public void init() throws Exception {

    try {
      BusinessObject s = BusinessObjectFactory.lookup("ws_student_external:DirStudentDataSearch");
      s.setValue("studata.trm_term_code", "FA10");
      bopm = new BusinessObjectPersistenceManager(s,"studata.name","/opt2/filerepository/bpel","dirsearch");
    } catch (Exception ex) {
       ex.printStackTrace();
    }

  }


    // Uses BusinessObjectPersistenceManager instead of array
    public List<Map<String,String>> getNames(String q) throws Exception {
                                        List<Map<String,String>> names = new ArrayList();
            Soundex sx = new Soundex();
            Vector v = bopm.getSelectListBy(sx.soundex(q));
            for (int i = 0; i < v.size(); i++) {
              BusinessObject b = (BusinessObject) v.elementAt(i);
              Map<String,String> m = new HashMap<String,String>();
              m.put("name", b.getStringValue("studata.stu_name"));
                                        names.add(m);
            }

                                        return names;
              }


}
```

Resources

- http://uxt.ucsd.edu/
- http://jquery.com/
- http://jqueryui.com/
- http://www.json.org/