knowledge14®

April 27–May 1, 2014 • Moscone Center • San Francisco, CA

Jelly 201: Demystify Jelly Scripting

Tyler Jones
Sr. Developer
ServiceNow

Chuck Tomasi
Sr. Services Enablement Program Manager
ServiceNow

servicenow®

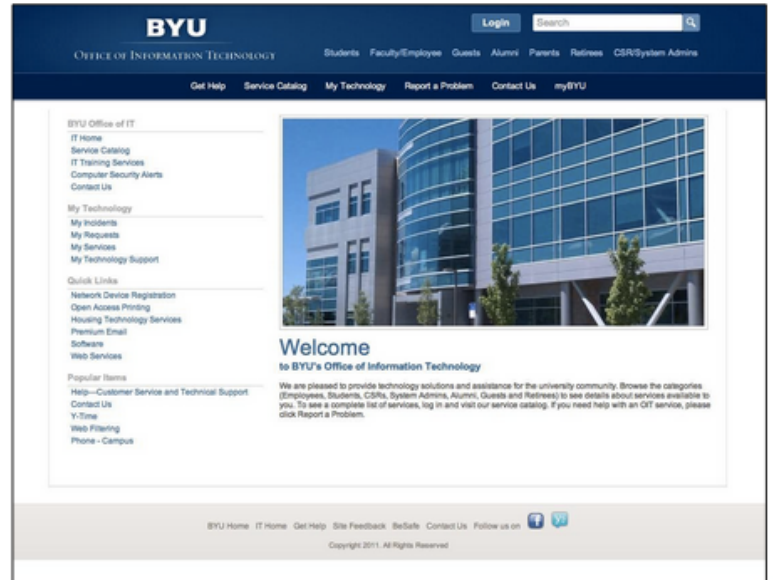
© 2014 ServiceNow All Rights Reserved

Tyler Jones

Sr. UI Developer



- Formerly ServiceNow CMS customer at BYU (2008-2011)
- ServiceNow since 2011
- Projects
  - CMS
  - Tablet
  - Mobile
  - Next Generation Desktop



© 2014 ServiceNow All Rights Reserved

Chuck Tomasi



Sr. Services Enablement Program Manager

- 30 years IT experience
- ServiceNow customer 2008-10
- First Innovation of the Year Award @ K10
- ServiceNow since 2010
- Lots of deployments, custom apps, and special projects
- Technical Best Practices owner
- Co-host of ServiceNow web series *TechNow*
- Other stuff:
  - Author, Podcaster, golfer, downhill skier, photographer, sci-fi geek, martial artist

© 2014 ServiceNow All Rights Reserved

Agenda

What is Jelly?

Basics of Jellly syntax and phases

Build a Jelly UI page

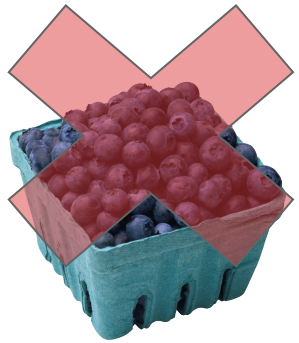
© 2014 ServiceNow All Rights Reserved

Jelly Basics

© 2014 ServiceNow All Rights Reserved

What is Jelly

- Java Elements originates from the Apache Foundation
  - http://commons.apache.org/jelly
- ServiceNow uses Jelly to process XML documents containing static HTML and special Jelly tags
- The output is HTML
  - You can't eat it



© 2014 ServiceNow All Rights Reserved

Why Learn Jelly?

- Deepen your understanding of the ServiceNow platform
- Jelly is used in UI Pages and UI Macros
  - There are ~180 UI Pages and ~210 UI Macros baseline
- But a word of caution…



© 2014 ServiceNow All Rights Reserved

Before You Dive In

- Treat baseline UI Pages and UI Macros as read-only
  - If you modify them, you will not receive upgrades to the original script
- Jelly != JavaScript
  - Debugging and maintaining Jelly script is challenging
  - Jelly developers need to be aware of ‘two-phase processing’

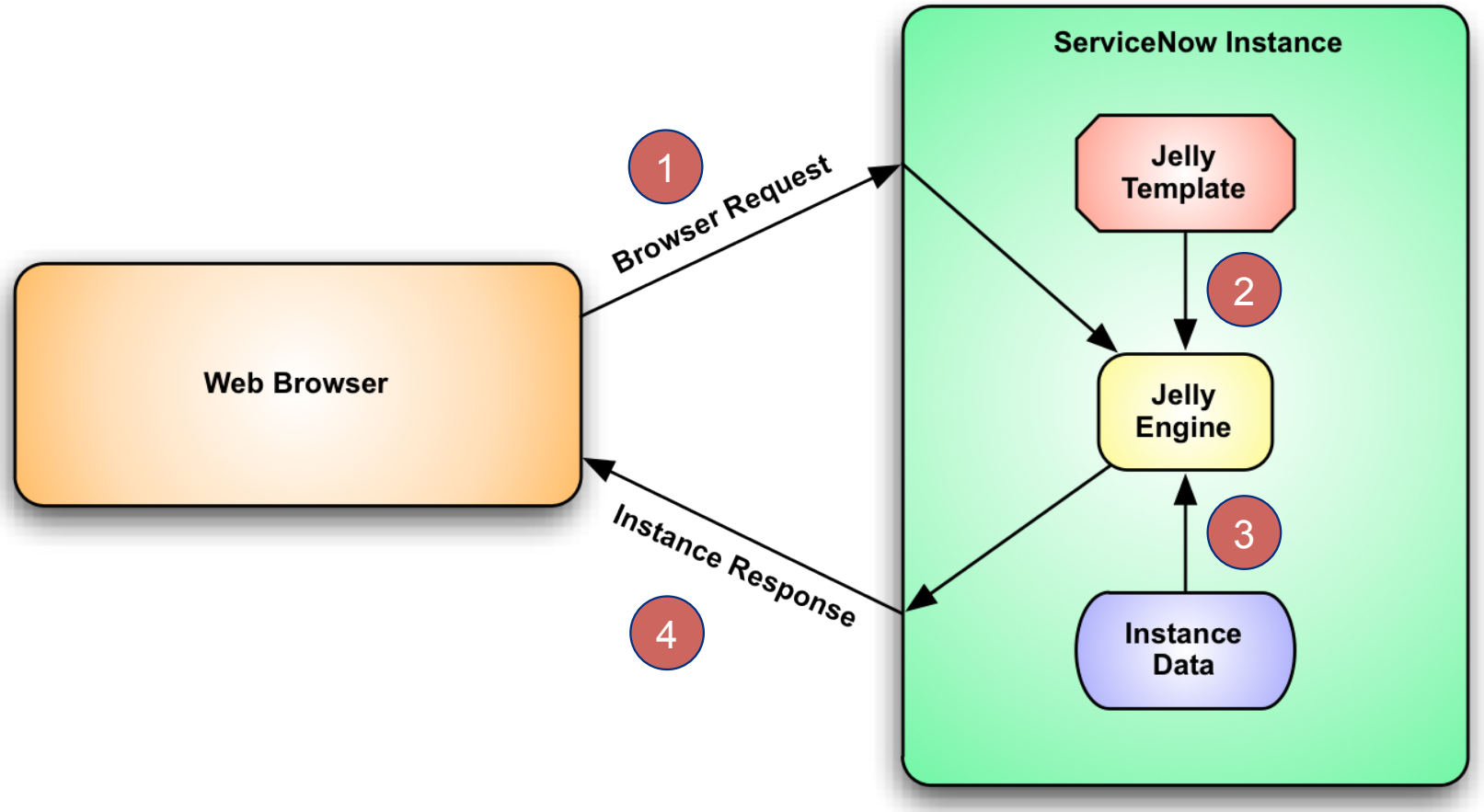


“Here be dragons”

© 2014 ServiceNow All Rights Reserved

Big Picture

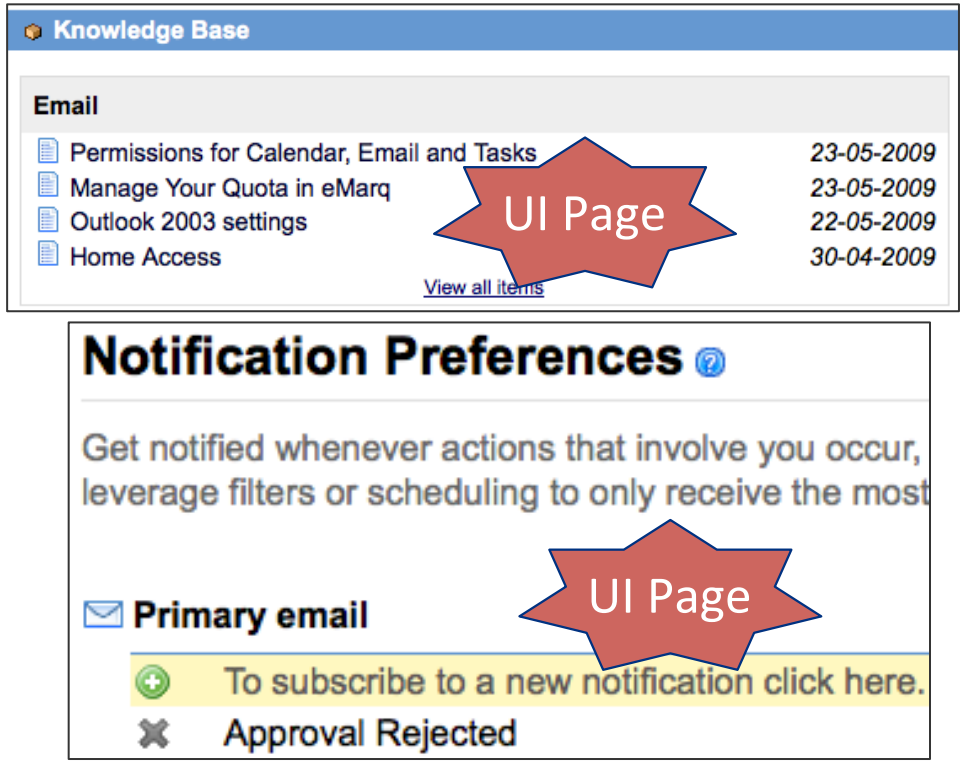
- Jelly tags define dynamic content that is processed to produce HTML

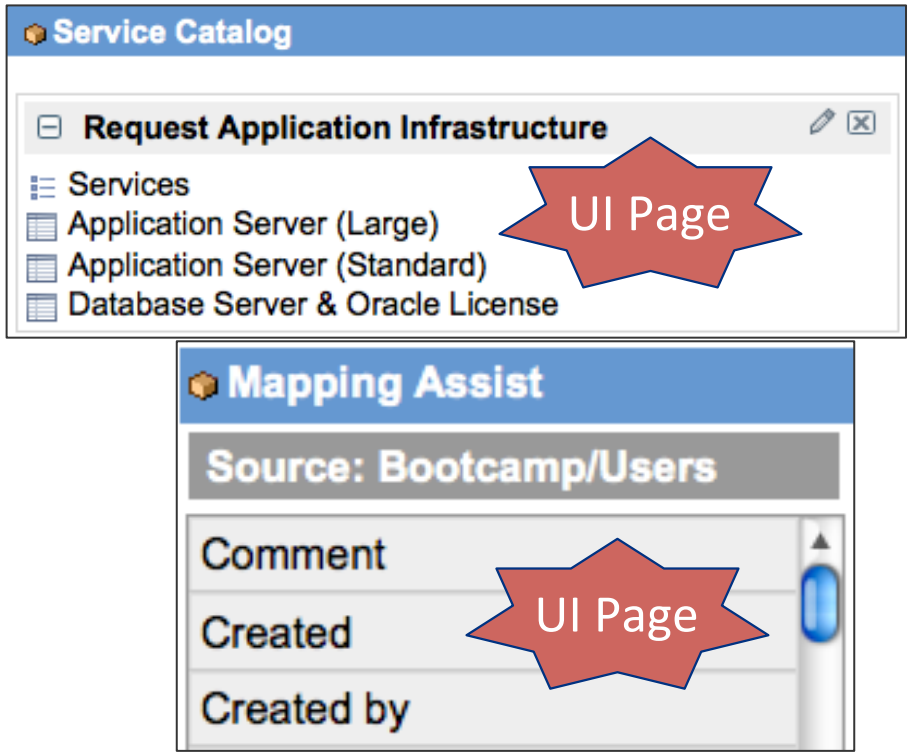


© 2014 ServiceNow All Rights Reserved

Jelly Context: UI Pages

- Fundamental building blocks of the user interface

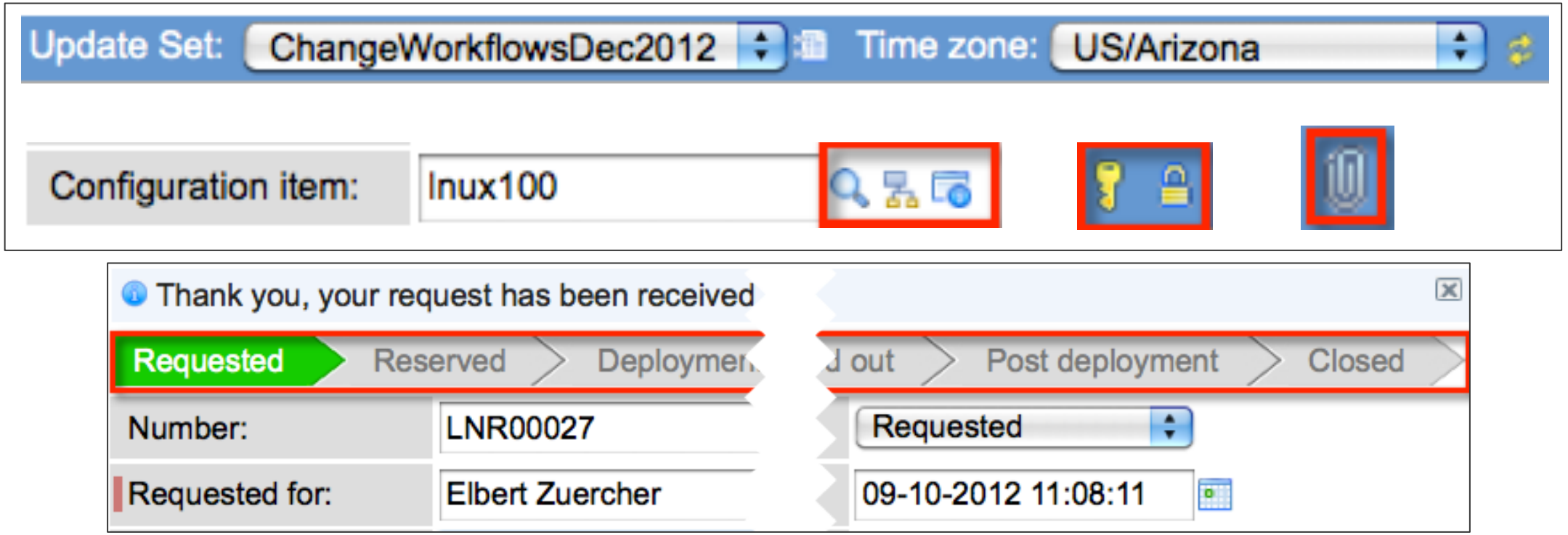




© 2014 ServiceNow All Rights Reserved

Jelly in Context: UI Macros

- Reusable UI components rendered using Jelly
  - Can be embedded in UI Pages



© 2014 ServiceNow All Rights Reserved

Jelly Tags

- Two kinds of tags
  - Core tags in the jelly:core namespace ('**j** tags')
  - ServiceNow custom tags in the glide namespace ('**g** tags')

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
```

- Examples:

Core Tags	Glide tags
<j:if>	<g:evaluate>
<j:set>	<g:breakpoint>
<j:while>	<g:macro_invoke>
<j:choose>	<g:call>

© 2014 ServiceNow All Rights Reserved

Jelly Phases

Original

```
<j:set var='jvar_first' value='Hello'/>
<j2:set var='jvar_second' value='World'/>

<p> ${jvar_first} </p>
<p> $[jvar_second] </p>
<p> Phases don't faze me </p>
```

After Phase 1

```
<j2:set var='jvar_second' value='World'/>

<p> Hello </p>
<p> $[jvar_second] </p>
<p> Phases don't faze me </p>
```

After Phase 2

```
<p> Hello </p>
<p> World </p>
<p> Phases don't faze me </p>
```

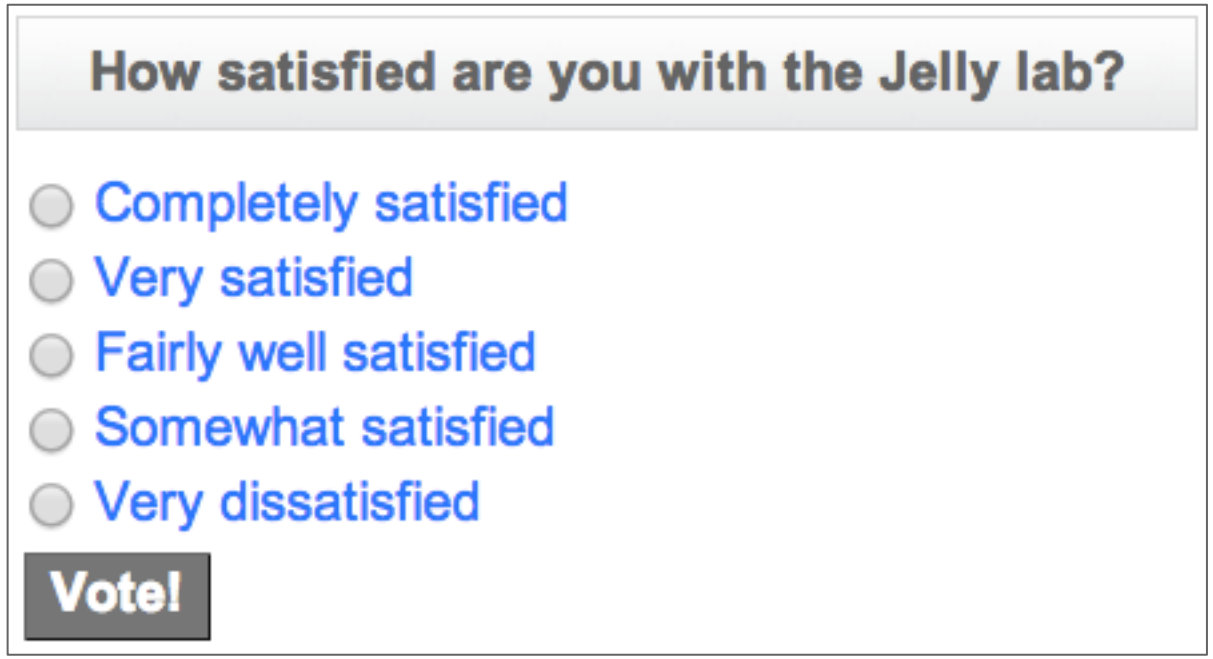
© 2014 ServiceNow All Rights Reserved

Let’s Build Something!

© 2014 ServiceNow All Rights Reserved

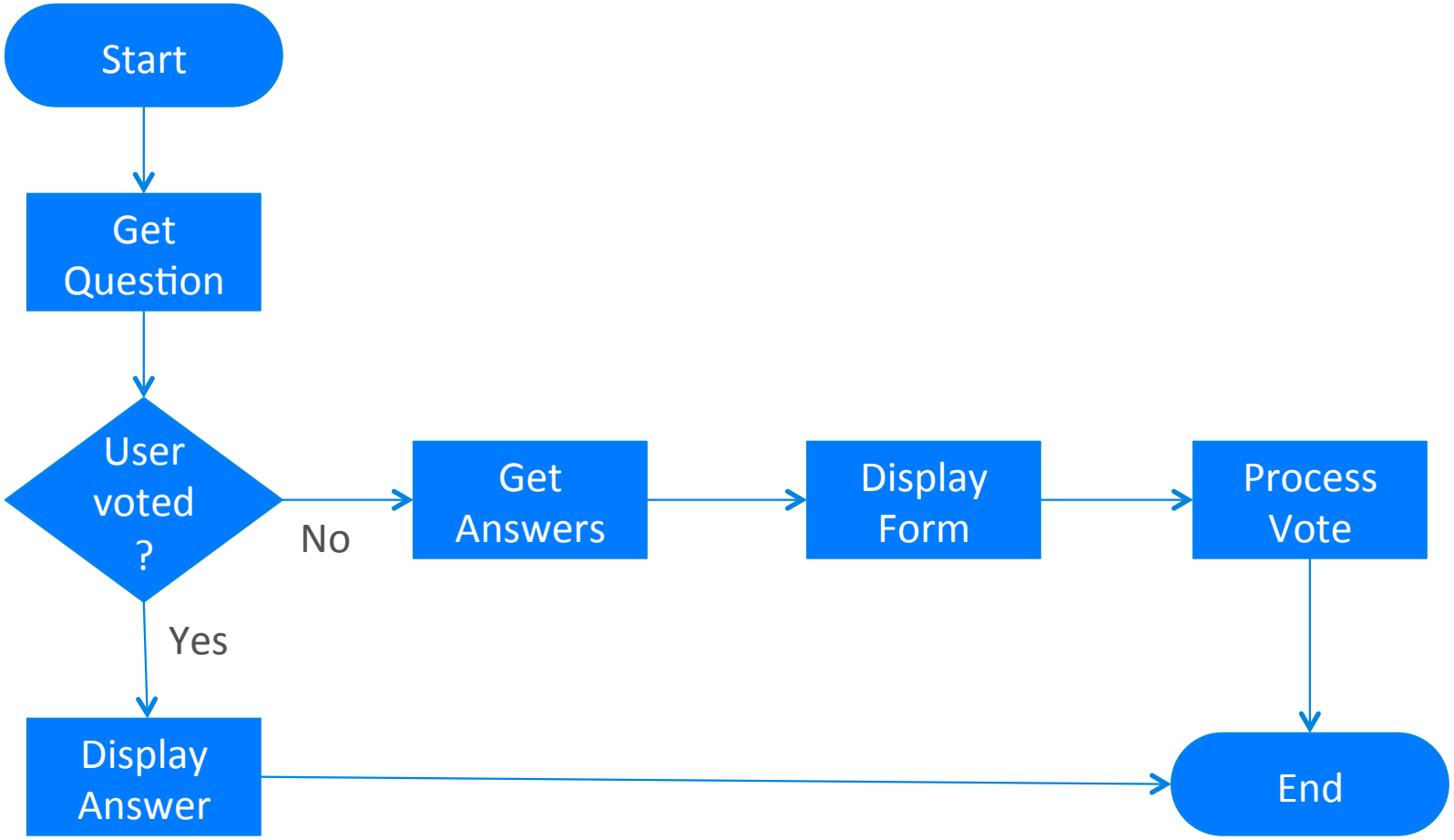
Polling App

- Ask a question
  - Display question and related answers
  - Wait for form submit
  - Save results on submit
- Once a user votes
  - Display their answer



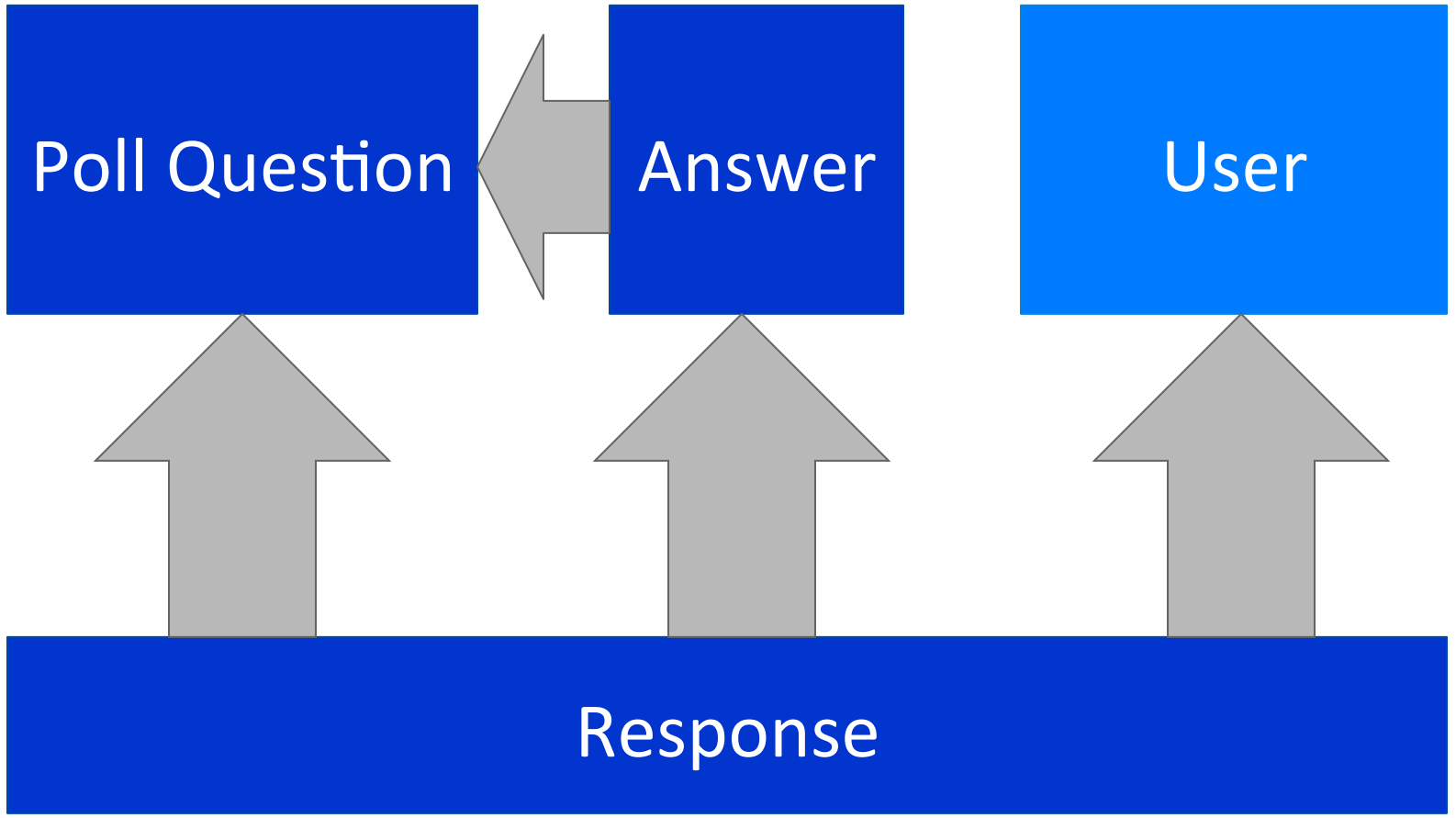
© 2014 ServiceNow All Rights Reserved

Process Flow



© 2014 ServiceNow All Rights Reserved

Data/Table Structure



© 2014 ServiceNow All Rights Reserved

Lab 1: Create a New UI Page

- Login to your instance as admin
- Navigate to **System UI** > **UI Pages**
- Click New
- Enter the following information:
  - Name: poll_view
  - Description: A poll manager UI Page
- The following appears in the HTML field:

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">

</j:jelly>
```

© 2014 ServiceNow All Rights Reserved

Lab 1: Create a UI Page

- Between the <j:jelly> and </j:jelly> tags, enter a simple string
- Click “Try It”

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null”>
Poll View 1.0<br />
</j:jelly>
```

- Click your browser “Back” button to return to your UI page definition

© 2014 ServiceNow All Rights Reserved

Lab 2: Get the User's ID and Test

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
Poll View 1.0<br />
<!--
    Get the current user's ID
    Concepts:
      Simple g:evaluate
      Passing a JavaScript value to Jelly variable
      Using the Jelly variable in a JEXL expression
  -->
<g:evaluate var="jvar_userid" expression="gs.getUserID;" />
Welcome user id: $[jvar_userid]<br />

    <!-- REPLACE THIS LINE WITH LAB 3 -->
</j:jelly>
```

© 2014 ServiceNow All Rights Reserved

Lab 3: Get an Active Poll ID

```
<g:evaluate var="jvar_poll_id">

  var p1 = new GlideRecord('u_poll');

  p1.addQuery('active', true);
  p1.orderByDesc('sys_updated_on');
  p1.query();

  if (p1.next()) {
    result = p1.getValue('sys_id');
  }

  result;
</g:evaluate>
The latest poll is: ${jvar_pollid}<br/>

 <!-- REPLACE THIS LINE WITH LAB 4 -->
```

© 2014 ServiceNow All Rights Reserved

Lab 4: Get the Question Text

Sidebar: Debugging

```
<g:evaluate>

  var p2 = new GlideRecord('u_poll');
  var questionText= '';

  if (p2.get(${jvar_poll_id}) {
    questionText = p2.getValue('u_question');

  } else {
    gs.print('>>>poll_view: Error reading question');
  }

</g:evaluate>
<div class="poll_question">${questionText}</div>

  <!-- REPLACE THIS LINE WITH LAB 5 -->
```

© 2014 ServiceNow All Rights Reserved

Lab 5: Has This User Voted Yet?

```
<g:evaluate>
  var response = new GlideRecord('u_poll_response');
  var result;

  response.addQuery('u_user', ${jvar_userid});
  response.addQuery('u_poll', ${jvar_poll_id});
  response.query();
  result = response.next();

  result;
</g:evaluate>

User has voted: ${jvar_has_voted}<br/>

  <!-- REPLACE THIS LINE WITH LAB 6 -->
```

© 2014 ServiceNow All Rights Reserved

Lab 6: If the User HAS Voted…

```
<j:if test="${jvar_has_voted eq true}">
  Your response was: ${response.u_answer.u_text}<br />
</j:if>

  <!-- REPLACE THIS LINE WITH LAB 7 -->
```

© 2014 ServiceNow All Rights Reserved

Lab 7: If the User has NOT Voted…

```
<j:if test="!${jvar_has_voted}">

  <g:evaluate>

    var ans = new GlideRecord('u_poll_answer');

    ans.addQuery('u_poll', jelly.jvar_poll_id);
    ans.orderBy('order');
    ans.query();
  </g:evaluate>
  We should have an answer object now ${ans.hasNext()}<br />

  <!-- REPLACE THIS LINE WITH LAB 8 -->

</j:if>
```

© 2014 ServiceNow All Rights Reserved

Lab 8: Ensure the Question has Answers

```
<!--
    Check if this is a question without answers
    Concepts:
      Conditional statement on a JavaScript variable/method
      gs.getMessage()

    <j:if test="${!ans.hasNext()}">
      <div class="poll_error">
        ${gs.getMessage('poll_no_answers_found');}
      </div><br />
    </j:if>

 <!-- REPLACE THIS LINE WITH LAB 9 -->
```

© 2014 ServiceNow All Rights Reserved

Lab 9: Construct the Form

```
<j:if test="${ans.hasNext()}">
  <g:ui_form>
    <input type="hidden" name="poll_id”
           value="${jvar_poll_id}" />
    <input type="hidden" name="user_id”
           value="${jvar_userid}" />

    <j:while test="${ans.next()}">
      <div class="poll_answer">
        <g:ui_radio name="poll_answer”
           value="${ans.sys_id}" text="${ans.u_text}" />
      </div>
    </j:while>
    <g:ui_button action="save">
       ${gs.getMessage('poll_vote_button')}</g:ui_button>

  </g:ui_form>
</j:if>
```

© 2014 ServiceNow All Rights Reserved

Lab 10: The Processing Script

```
var pr = new GlideRecord('u_poll_response');

if (!JSUtil.nil(poll_id) && !JSUtil.nil(user_id)) {
   pr.initialize();
   pr.u_poll = poll_id;
   pr.u_user = user_id;
   pr.u_answer = poll_answer;
   pr.update();

}
```

© 2014 ServiceNow All Rights Reserved

Best Practices

- Document your work
- Avoid dynamic JEXL inside evaluate tags
- Use descriptive variable names
- Use appropriate indentation

© 2014 ServiceNow All Rights Reserved

Top Takeaways

1. Most of what you need can be done in Phase 1

2. Understand Jelly vs. JavaScript variables

3. Use best practices (no dynamic JEXL in evaluate tags)

© 2014 ServiceNow All Rights Reserved

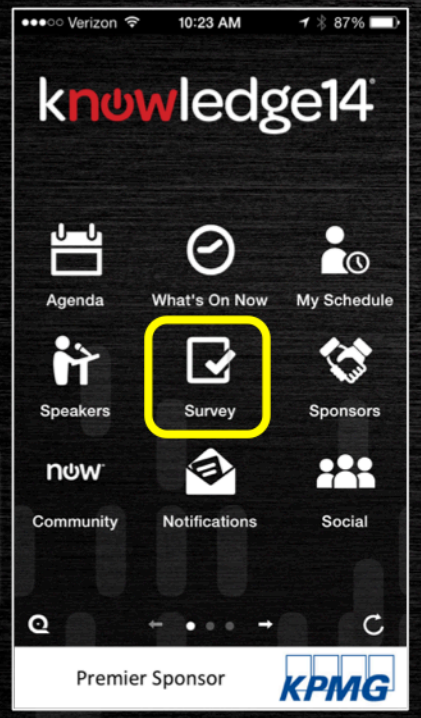
Additional Resources

- https://support.service-now.com
  - Wiki
  - Community
  - Technical Support
  - Knowledge Base
- TechNow videos
  - http://youtube.com/servicenowdemo



© 2014 ServiceNow All Rights Reserved

Tell us what you think!



Click on the survey link in your Knowledge14 mobile app. Help us make Knowledge 15 even more valuable!

Thank You

Tyler Jones

Sr. Developer

ServiceNow

tyler.jones@servicenow.com

Chuck Tomasi

Sr. Services Enablement Program Manager

ServiceNow

chuck.tomasi@servicenow.com

Want a copy of this deck?

Sign into the BuzzHub to unlock all the slides and lab guides from Knowledge14 sessions

