Fiz: A Component Framework for Web Applications

John Ousterhout
Stanford University
My Background

1980 - U.C. Berkeley
- VLSI CAD: Caesar, Magic, Crystal, etc.

1990
- Sprite OS
- Log-Structured FS
- Tcl/Tk

2000 - Sun Scriptics
- Tcl Tools
- Electric Accelerator

2010 - Stanford
- Electric Cloud
- Fiz
- Electric Commander
History of the Web

- Server-browser disconnect
- Weak features for interaction
- Browser incompatibilities

DOM specification
Browser compatibility
AJAX, etc.

Documents
Forms
Applications!

Web evolution
The Web is Changing Everything

Discovering the potential:
- New applications
- 100-1000x scale
- New development style
- New approach to deployment

Realizing the potential:
- New models of computation (EC2)
- New storage (BigTable)
- New algorithms (MapReduce)
- New languages
- New frameworks
- New approaches to software development
Fiz Introduction

● **The problem:**
  - Too hard to develop interactive Web applications
  - Existing frameworks too low-level

● **The solution:**
  - Raise the level of programming: don’t write HTML!
  - Create applications from high-level reusable components

● **Fiz:**
  - Framework for creating components for Web applications
  - Library of built-in components
  - Goal: create community around component set

● **Research challenge:**
  - Design powerful building blocks that hide Web complexity
- Think about page structure, not HTML
- Encourage creation of higher-level components
**Templates (Rails)**

```html
<table class="TableSection" cellspacing="0">
  <tr class="header">
    <td>Name</td>
    <td>Student Id</td>
    <td>Graduation Year</td>
    <td>GPA</td>
  </tr>
  <% for student in @students -%>
  <tr class="<%= cycle('even', 'odd') %>">
    <td><%= link_to(student.last + ', ' + student.first, student) %></td>
    <td><%= student.id %></td>
    <td><%= student.grad %></td>
    <td><%= student.gpa %></td>
  </tr>
  <% end -%>
</table>
```

**Code to generate dynamic content**

**Variable substitution**

**HTML**

<table>
<thead>
<tr>
<th>Name</th>
<th>Student Id</th>
<th>Graduation Year</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anderson, Alice</td>
<td>2149821</td>
<td>2009</td>
<td>3.6</td>
</tr>
<tr>
<td>Benson, Bob</td>
<td>2147322</td>
<td>2010</td>
<td>2.9</td>
</tr>
<tr>
<td>Collins, Carol</td>
<td>3990714</td>
<td>2012</td>
<td>3.2</td>
</tr>
<tr>
<td>Dawson, David</td>
<td>4027333</td>
<td>2009</td>
<td>3.8</td>
</tr>
<tr>
<td>Evans, Ellen</td>
<td>4019329</td>
<td>2012</td>
<td>3.7</td>
</tr>
</tbody>
</table>
<table class="TableSection" cellspacing="0">
  <tr class="header">
    <td>Name</td>
    <td>Student Id</td>
    <td>Graduation Year</td>
    <td>GPA</td>
  </tr>
  <% for student in @students -%>
  <tr class="<%= cycle('even', 'odd') %>">
    <td><%= link_to(student.last + ', ' + student.first, student) %></td>
    <td><%= student.id %></td>
    <td><%= student.grad %></td>
    <td><%= student.gpa %></td>
  </tr>
  <% end -%>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Student Id</th>
<th>Graduation Year</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anderson, Alice</td>
<td>2149821</td>
<td>2009</td>
<td>3.6</td>
</tr>
<tr>
<td>Benson, Bob</td>
<td>2147322</td>
<td>2010</td>
<td>2.9</td>
</tr>
<tr>
<td>Collins, Carol</td>
<td>3990714</td>
<td>2012</td>
<td>3.2</td>
</tr>
<tr>
<td>Dawson, David</td>
<td>4027333</td>
<td>2009</td>
<td>3.8</td>
</tr>
<tr>
<td>Evans, Ellen</td>
<td>4019329</td>
<td>2012</td>
<td>3.7</td>
</tr>
</tbody>
</table>
new TableSection(
    new Dataset("request", "getStudents"),
    new Column("Name", new Link(
        "@last, @first", "show?id=@id")),
    new Column("Student Id", "@id"),
    new Column("Graduation Year", "@grad"),
    new Column("GPA", "@gpa")
)
new TableSection(
    new Dataset("request", "getStudents"),
    new Column("Name", new Link("@last, @first", "show?id=@id")),
    new Column("Student Id", "@id"),
    new Column("Graduation Year", "@grad"),
    new Column("GPA", "@gpa"))
### Fiz TableSection

```java
new TableSection(
    new Dataset("request", "getStudents"),
    new Column("Name", new Link(
        "@last, @first", "show?id=@id"),
    new Column("Student Id", "$id"),
    new Column("Graduation Year", "@grad"),
    new Column("GPA", "@gpa"))
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Student Id</th>
<th>Graduation Year</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anderson, Alice</td>
<td>2149821</td>
<td>2009</td>
<td>3.6</td>
</tr>
<tr>
<td>Benson, Bob</td>
<td>2147322</td>
<td>2010</td>
<td>2.9</td>
</tr>
<tr>
<td>Collins, Carol</td>
<td>3990714</td>
<td>2012</td>
<td>3.2</td>
</tr>
<tr>
<td>Dawson, David</td>
<td>4027333</td>
<td>2009</td>
<td>3.8</td>
</tr>
<tr>
<td>Evans, Ellen</td>
<td>4019329</td>
<td>2012</td>
<td>3.7</td>
</tr>
</tbody>
</table>
new TableSection(
    new Dataset("request", "getStudents"),
    new Column("Name", new Link("@last, @first", "show?id=@id")),
    new Column("Student Id", "@id"),
    new Column("Graduation Year", "@grad"),
    new Column("GPA", "@gpa"))

Generate <a> element URL

Name | Student Id | Graduation Year | GPA
--- | --- | --- | ---
Anderson, Alice | 2149821 | 2009 | 3.6
Benson, Bob | 2147322 | 2010 | 2.9
Collins, Carol | 3990714 | 2012 | 3.2
Dawson, David | 4027333 | 2009 | 3.8
Evans, Ellen | 4019329 | 2012 | 3.7
Sections

- Encapsulate particular styles of information display and user interaction:
  - Table
  - Form
  - Navigation tabs
  - …

- Generate HTML

- Handle user interactions

- May include Javascript, AJAX

- Example:

  ```javascript
  new TreeSection(new Dataset("request", "demo.tree"))
  ```
public class StudentsInteractor extends Interactor {
    public void showAll(ClientRequest cr) {
        cr.getHtml().setTitle("Current Students");
        Section sections[] = {
            new TemplateSection("<h1>Your University Online</h1>
            new TabSection(...);
            new TemplateSection("<h1>Current Students</h1>
            new TableSection(...);
            new TemplateSection("<h1>Enter New Student</h1>
            new FormSection(...);
        };
        cr.showSections(sections);
    }
}
Data Managers

URLs → Fiz Dispatcher

Interactor → Interactor

Sections
- Tabs
  - Inventory
  - Orders
  - Accounts
  - Shippers

- Table
  - Form

- Tree

Front End
- Data Requests

Back End
- Data Manager
  - Enterprise Application
  - SQL Database
  - Remote Feed
Data Managers

- Provide access to particular kinds of data:
  - Relational database
  - Remote data feed
  - Existing application (local or remote)
  - Excel spreadsheet

- Most frameworks:
  - ORM specialized for single RDBMS data source
  - Synchronous data accesses

- Fiz:
  - Supports a variety of data sources
  - Asynchronous accesses for higher performance
Fiz vs. Rails Templates

![Graph comparing lines of code for Fiz and Rails in different components: Table, Form, Tabs. The graph shows the number of lines of code for each component and the efficiency comparison.]

<table>
<thead>
<tr>
<th>Component</th>
<th>Fiz</th>
<th>Rails</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>34</td>
<td>6</td>
</tr>
<tr>
<td>Form</td>
<td>12</td>
<td>113</td>
</tr>
<tr>
<td>Tabs</td>
<td>11</td>
<td>10.3x</td>
</tr>
</tbody>
</table>

- **Rails Javascript**: 6.4x
- **Rails CSS**: 5.7x
- **Rails Template**: 10.3x

### Student Database

<table>
<thead>
<tr>
<th>Name</th>
<th>Student ID</th>
<th>Graduation Year</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anderson, Ali</td>
<td>2149621</td>
<td>2009</td>
<td>3.6</td>
</tr>
<tr>
<td>Benson, Bob</td>
<td>2147322</td>
<td>2010</td>
<td>2.9</td>
</tr>
<tr>
<td>Collins, Carol</td>
<td>3990714</td>
<td>2012</td>
<td>3.2</td>
</tr>
<tr>
<td>Dawson, David</td>
<td>4027333</td>
<td>2009</td>
<td>3.8</td>
</tr>
<tr>
<td>Evans, Ellen</td>
<td>4019329</td>
<td>2012</td>
<td>3.7</td>
</tr>
</tbody>
</table>

Last name: [ ]
First name: [ ]
Student ID: [ ]
Graduation year: [ ]
GPA: [ ]
**Status**

- January 2008: started implementation
- Fall 2008: started first application (Fiz community Web site)
- **Built-in sections:**
  - Table
  - Form
  - Tabs
  - Tree
  - Template
  - Compound
- **Planning first open-source release Summer 2009**
  - Goal: create open-source community
- **Interested in industrial collaboration**
Potential Impact

- Faster application development
- Better applications
- More code reuse
- Components hide Web complexity:
  - Security issues
  - Interaction styles
  - Performance issues
  - Mobile device differences
Open Questions

Mostly related to scalability:

- How many useful components are there?
- Can smaller components be composed into larger components?
- How well will Fiz work for large applications?
- Do components capture a significant fraction of application functionality?
- Are the components customizable enough?
- How much additional complexity does Fiz introduce?
Conclusions

- Web applications growing in importance; need to improve development process
- Traditional template usage is a bad idea
  - Works against components
- Component orientation results in different framework features
- Preliminary evidence:
  - Components can encapsulate interesting functionality
  - Components can reduce application front-end 5-10x
- Lots more work needed to validate Fiz, find limits of component frameworks for the Web
Questions?
Web Development Complexities

- Distributed (browser vs. server)
- Multiple languages and technologies
- Browser incompatibilities
- Scalability: 100-1000x vs. pre-Web applications
- Customizability
- Security issues
Current Framework Facilities

- **Templates**
  - Mix HTML and code for dynamic content

- **Object-relational mapping (ORM)**
  - Rows from database tables appear as language objects
  - Interactions with database hidden

- **Sessions**
  - Manage series of requests from the same browser

- **Integrations, etc.**

**Too low-level: tools for HTML generation**