CS 635 Advanced Object-Oriented Design & Programming
Spring Semester, 2004
Doc 17 Model-View-Controller part 2
Contents

References

Patterns of Enterprise Application Architecture, Folwer, 2003, pp 330-386

Core J2EE Patterns: Best Practices and Design Strategies, 2nd, Alur, Crupi, Malks, 2003

Transform View

A view that processes domain data elements by element and transforms them into HTML

Given a domain object, MusicAlbum, how to generate a web page for the object?

- Use Template View
- Convert object into html

Converting object into html

One could add toHtml to the object

```
MusicAlbum ragas = new MusicAlbum.find("Passages");
String html = ragas.toHtml();
```

- Domain object is coupled to view language
- Provides only one way to display object

Better use XML and XSLT

- Convert domain object to XML
- Use XSLT to convert XML into HTML

Now we can produce many different views of the object without changing the object

Transform View Verses Template View

Template View

More tools support

No language to learn

Transform View

Easier to avoid domain logic in view

Testing can be done without Web server

Easier to make global changes to Web site

Context Object

Problem

Avoid using protocol-specific system information outside its relevant context

Forces

- You have components and services that need access to system information
- Decouple application components & services from protocol specifics
- You want to expose only relevant APIs within a context

Example

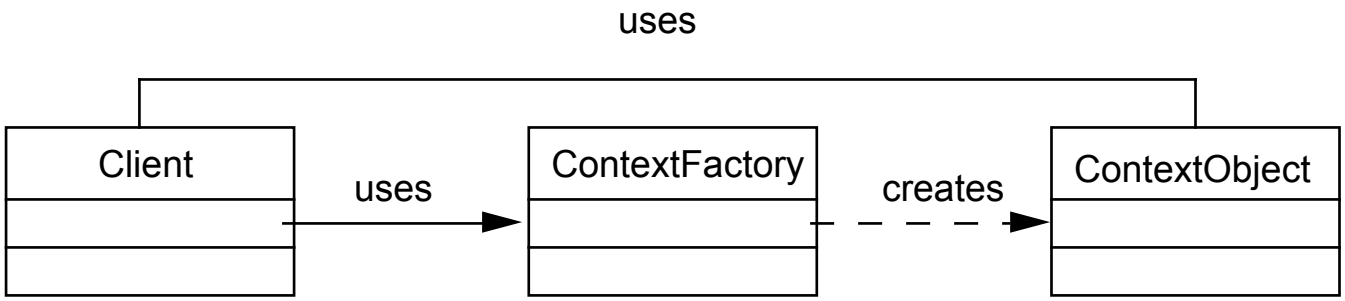
```
public class AlbumController extends ActionServlet {

  public void doGet(HttpServletRequest request,
      HttpServletResponse response)
      throws IOException, ServletException {
```

HttpServletRequest & HttpServletReponse contain protocol specific information and API

Solution

Use Context Object to encapsulate state in a protocol-independent way to be shared throughout the application



Application Controller

Action management – map a request to an action that handles request

View management – locate and use the correct view

Problem

Centralize & modularize action and view management

Forces

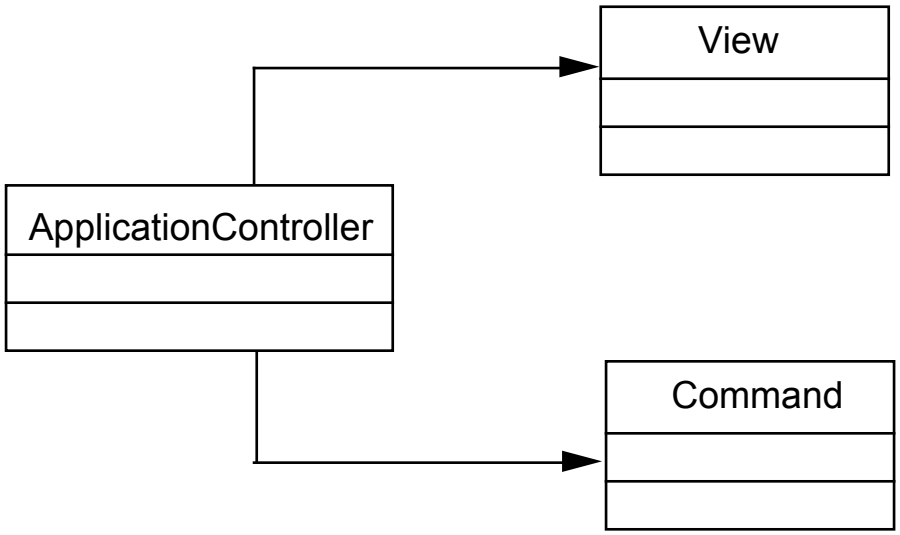
- Want to reuse action & view management
- Improve request-handling extensibility
- Improve code modularity & maintainability

Solution

Use an Application Controller to centralize retrieval and invocation of commands and views

A centralized point for handling screen navigation and the flow of the applications

Structure



When to use it

When web pages should be visited in particular order

Different views are used depending on the state of objects

Continuation-Based Web Servers

- Cocoon Flow (Java +Javascript+XSLT)
http://wiki.cocoondev.org/

- Seaside (Smalltalk)
http://www.beta4.com/seaside2/

- PLT Scheme
http://www.plt-scheme.org/

- Sisc (Scheme in Java)
http://sisc.sourceforge.net/

Seaside

http://beta4.com/seaside2/

Free with source
- Squeak Smalltalk
- VisualWorks Smalltalk
- Ruby (called Borges)

http://segment7.net/projects/ruby/borges/index.html

Runs on:
- Unix
- MacOS
- Linux
- Windows

Runs standalone or behind:

- Apache
- Microsoft IIS
- Netscape Server
- Others (CGI, FastCGI)

Example

ContinuationExample Subclass of WAComponent

```
renderContentOn: html
   html title: 'Continuation Example'.
   html heading: 'The First Page'.
   html anchorWithAction: [self tryMe] text: 'Start'

tryMe
   | name count message |
   name := self request: 'Your name'.
   count := 1.
   message := name , ' ready to stop yet? '.
   [self confirm: message , count printString]
      whileFalse: [count := count + 1].
   self inform: 'Good bye'
```

Java Psuedocode

```
renderContentOn(WAHtmlRenderer html) {
   html.title(”Continuation Example”);
   html.heading(”The First Page”);
   html.anchorWithAction( tryMe, “Start”);
}

tryMe() {
   String name = request(”Your name”);
   int count = 1;
   String message = name + “ ready to stop yet”;
   while ( !confirm( message + count) ) {
      count = count + 1;
   }
   inform( “Good bye”);
}
```

Wafer Project

- Simple web blog implementation
- Used to compare Java Web frameworks
- http://www.waferproject.org
- Implemented in Seaside by Todd Blanchard

Wafer Web Blog Implementation

Metric	Java Struts	Seaside
Development Time	1-2 weeks	~6 hours
Size	4 MB	26 KB
Files	155	1
Source Size	159 KB	26 KB
Classes	36	12