Baby's First Java Web Server

CPE 305-01

James Pearson

2011-11-30

Framework Specification

This proposal concerns the implementation of a basic MVC Java web framework.

As with graphical applications, the standard architecture pattern for web applications is Model-View-Controller (MVC). A basic diagram looks something like this:

```
           ------------
|-------> | web server |
|          ------------
|               |
|               v
|  -----------------------------
| |  ------------               |
| | | middleware |              |
| |  ------------               |
| |        |                    |
| |        v                    |
| |  -------------              |
| | | URL routing |             |
| |  -------------              |
| |        |                    |
| |        v                    |
| |  ------------      -------  |
| | | Controller | -> | Model | |
| |  ------------  <-  -------  |
| |        |                    |
| |        v                    |
| |      ------                 |
--|     | View |                |
  |      ------                 |
   -----------------------------
```

That is, a web server handles the socket connections and input, passes data along to the framework through some predetermined specification. The framework first runs requests through optional, pluggable middleware - common examples are caching, gzipping, and authentication. The request is then passed along to a URL router, which uses a set of user-defined rules to determine, based on the requested URL, which controller method should be called. The controller pulls data from any necessary models, then passes the retrieved data on to the view, which uses its parameters to produce a response of some sort (usually HTML). This response is passed back to the web server, which forwards it on to the user.

Application 1 - Simple Pony Image Gallery API

"My Little Pony: Friendship is Magic" (MLP:FiM) is a currently-airing children's television show that promotes being awesome. 20-something male followers of the show refer to themselves as "bronies". To assist bronies with creating fabulous applications, the service will provide an API to query a database of pony images, with filtering capabilities on the characters pictured.

Example request:
```
/images/tags/applejack+twilight_sparkle-pinkie_pie
```

Example response:
```
[
  {
    "url": "/static/yg2cX.jpg",
    "md5": "b35ada278327e3bdf128dbd9b7edd345",
    "height": 900,
    "width": 1440,
    "tags": ["applejack", "apple_blossom", "twilight_sparkle"]
  },
  {
    "url": "/static/gea71.jpg",
    "md5": "43ac5915b25ef74e51780a054f76cb47",
    "height": 900,
    "width": 1440,
    "tags": ["applejack", "twilight_sparkle"]
  }
]
```

Application 2 - Favorite Pony Image Finder

Find a user's favorite pony image in a similar fashion as Pandora. Upon first visiting the site, the user is presented with an image and three options - downvote, next, upvote. Downvotes will show less images with the tags of the voted image, while upvotes will show more similar images. Next advances to another image without changing the user's calculated image preferences.

Because dogfooding is a good idea, this application will use the API presented above as "Application 1".

Application 3 - iFixit Device/Guide Browser (Boring Alternative)

Use the iFixit public API[0] to obtain a list of devices known to the site, then display them in a user-consumable format. For each device, a link should be provided to another page that displays more information about the device and a list of guides related to the device.

[0]: http://www.ifixit.com/api/0.1/doc

Design Rationale

Issue #1

Common Gateway Interface (CGI) is a well-established protocol for communicating between programming languages and web servers. Should Baby's First communicate via CGI?

Alternative A

Input and output data according to the CGI specification.

Advantages: Baby's First can be used with any web server that supports CGI.

Disadvantages: CGI supports a wider range of features than we want to support in a simplistic framework like Baby's First, thus complicating the design.

Alternative B

Use a custom protocol.

Advantages: A precise matching to Baby's First's feature set.

Disadvantages: Custom web server software must be used.

Decision

Alternative B. For a learning project such as this, simplicity is preferable over interoperability. The custom web server 11 lines of shell and the standard Unix utility `netcat`.

Issue #2

Baby's First needs a way to allow the user to specify a mapping between URLs and controllers.

Alternative A

Call a method in `BabysFirst` with the URL as a parameter, then instruct the user to extend the class and override the method.

Advantages: Uses standard Java operations. Allows the user full control over how URLs should be mapped.

Disadvantages: Gives the user no assistance in parsing and mapping the URL.

Alternative B

Utilize a static mapping of URLs to methods in a user-provided XML file.

Advantages: XML is familiar to most Java developers. A number of XML-parsing libraries are available that would make the parsing implementation fairly simple. Users don't have to parse URLs.

Disadvantages: URLs with GET parameters would need to be specified for every possible parameter value (inflexible). XML is ugly and verbose.

Alternative C
Use user-provided regular expressions to map URLs to controllers, a la Django.

Advantages: The user only needs to parse URLs by way of regular expressions, not through Java's painful string parsing. Flexible.

Disadvantages: Users need to learn to use basic regular expression syntax. Baby's First parsing will be slightly more complicated than standard string matching.

Alternative D
Implicitly redirect to controllers based on the URL pattern itself, a la Ruby on Rails.

Advantages: The user doesn't need to write any URL-mapping rules.

Disadvantages: Implicit mappings can be frustrating for new users. Users may resort to poor class structure and naming to get the URLs they want.

Decision
Alternative C. The downside is that the user needs to learn regular expressions, which they should learn anyways. Powerful without being very complicated.

Issue #3

Should Baby's First include a custom Object-Relational Mapper (ORM)?

Alternative A
Write a custom ORM.

Advantages: Setup can be minimized by using defaults suitable for a simple educational project. No need for the user to install other libraries.

Disadvantages: Increased complexity of implementation. Decreased flexibility. Users cannot transfer knowledge of other ORMs to Baby's First, or use much of the knowledge gained from Baby's First in other projects.

Alternative B
Let the user select and configure a third-party ORM, if they so desire.

Advantages: If the user doesn't need persistence, they don't need to do anything extra. Experience gained will be useful in other projects. Large community support. Available tutorials.

Disadvantages: Most available Java ORMs (particularly Hibernate) are complex, requiring a substantial amount of developer time to learn how to use them properly. The user must manage an extra library.

Decision
Alternative B. Many simple applications will not need persistence at all, and those that do can achieve their goals using simple serialization.

Issue #4

Should Baby's First include a templating language?

Alternative A

Include a custom templating language.

Advantages: Can be simplified to just what Baby's First users need.

Disadvantages: Adds more implementation work. Limits the user in what they can do.

Alternative B

Include a third-party templating language and custom wrappers around it.

Advantages: Wrappers can make common use-cases easy. Users can still use the more powerful features of the language by calling it directly.

Disadvantages: Syntax of the templates themselves may be unnecessarily complicated.

Alternative C

Let the user handle all templating themselves.

Advantages: Full flexibility.

Disadvantages: Users must choose and configure a templating language. Full complexity of whatever language they choose.

Decision

Alternative B. A good tradeoff between simplicity and flexibility.