Ruby on Rails

What is Ruby on Rails?
- Web Framework for DB driven web applications.
- Runs on Ruby language.
- Utilizes Ajax.
- Model-View-Controller style connectivity between the data storage, business logic, and interface.

The Learning Curve
- Steep initial learning due to learning both new language and framework simultaneously.
- Ruby is “Reflectively Object Oriented.”
- Rails has its own language.

Support
Although a new framework, there are many resources available to help learn, develop and troubleshoot Ruby on Rails projects
- Official website provides complete API lookup
- Recommended texts
- Official on-line tutorials, how-tos, and manuals
- Official mailing lists, wiki, message board and IRC channels
- GOOGLE! User tips, tricks and guides
There is a vast wealth of knowledge available as this list does not include resources specific to the Ruby programming language.
**Maintainability**

- Ruby on Rails produces concise easy to read code
- Code Generation Mechanism
- User freedom not to rely on this mechanism

**OS/DBMS/Web Server Compatibility**

- Operating Systems
  - Windows
  - Mac OS X
  - Linux

- Database Management Systems
  - Oracle
  - MySQL
  - PostgreSQL
  - Others

- Web Servers
  - Apache
  - Lighttpd
  - Mongrel
  - Others
  - FastCGI recommended

- Support for common platforms makes it easier to use Rails with existing systems
- Rails follows 'convention over configuration' so existing databases that don't follow the Rails conventions will require some extra work to use with Rails applications

**Web Service Support**

- ActionWebService module
  - SOAP
  - XML-RPC
  - REST
  - Web service scaffold exists to help with code generation

- Representational State Transfer (REST)
  - Based on the concept of remote resources
  - Uses standard HTTP commands to perform basic operations on resources (CRUD operations)
  - HTTP put ➔ Create
  - HTTP get ➔ Read
  - HTTP post ➔ Update
  - HTTP delete ➔ Delete

- Rails is beginning to provide more support for REST, which mirrors the common actions performed on databases

**Integration with other Technologies**

- Ajax
  - Ruby on Rails includes the Prototype and script.aculo.us Javascript libraries
  - Scaffolding helps with code generation

- Java
  - Access to Java classes from Ruby
  - JRuby
    - Ruby interpreter written in Java
  - Ruby Java Bridge (RJB)
    - Ruby library, uses standard Ruby interpreter

- Flex/Flash
  - WebORB for Ruby on Rails
    - Plugin that exposes Ruby classes to clients using Flex and Flash remoting
**Comparison with other Web Frameworks**

- **Features**
  - 34 frameworks, 7 are shown to support all listed features, with Ruby on Rails among those 7

- **Performance**
  - Neither fastest, nor slowest in terms of performance compared to other frameworks
  - Server configuration is important

- **Development Time**
  - Justin Gehtland (author of "Springs: A Developer's Notebook") compared using Java Springs to Ruby on Rails
    - Rails required about 1/3 as many lines of code and 1/10 as many lines of configuration
  - Sean Kelly (from the Nasa Jet Propulsion Lab) also did some comparisons of using JSP/servlets/Hibernate as opposed to Rails and some Python frameworks
    - Rails and Python frameworks required far fewer lines of code/configuration, with some frameworks requiring fewer lines than Rails
    - Rails seemed to be more difficult to use than other frameworks

**Case Studies**

Ruby on Rails is designed to be easily used by companies of any size for any purpose.

**Companies using Ruby on Rails:**
- Penny Arcade
- US Yellow Pages
- Twitter
- Amazon
- 43Things
- Blogger
- Many Others...

**Types of projects**
- E-Commerce
- Remote file storage
- Blogs
- Forums
- E-zines
- Project Management
- Web Comics
- “You name it”...

**Examples**

- And now for an example or two...

- **Example 1**
  - Steps:
    - $>rails RubyExample
    - (add username/pass to dbconfig file)
    - $>./RubyExample/script/generate model userProfile
    - $>./RubyExample/script/generate controller example1 index
    - add line: scaffold :userProfile to RubyExample/app/controllers/example1_controller.rb
    - Done.

- **Example 2**
  - Steps:
    - $>./RubyExample/script/generate controller example2 index
    - add lines to RubyExample/app/models/user_profile.rb:
      - validates_presence_of :firstName (etc... for others)
    - $>./RubyExample/script/generate scaffold userProfile example2
    - This generates all of the code, which was being handled by the scaffold, into files.
    - Now I can modify the html code templates. And lets see what we get!