Smart Presentation

Client-Server technologies and protocol

Google Protocol Buffers

- language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler
- Binary data
- Data structures (called "messages") and services are defined in the Proto Definition file (.proto) which is then compiled with protoc
- The generated class provides getters and setters for the fields that make up a protocol buffer and takes care of the details of reading and writing the protocol buffer as a unit

Google Protocol Buffers

- Message example (.proto file):

```
package tutorial;
option java_package = "com.sampullara.jaxrsprotobuf.tutorial";
option java_outer_classname = "AddressBookProtos";
message Person {
  required string name = 1;
  required int32 id = 2;
  optional string email = 3;
  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }
  message PhoneNumber {
    required string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }
  repeated PhoneNumber phone = 4;
}
message AddressBook {
  repeated Person person = 1;
}
```

Jersey JAX-RS Web Server

- The simplicity of **RE**presentational **S**tate **T**ransfer (REST), an architectural style for accessing information on the web, has made it a popular way for developers to access services. In the REST architectural style, information on the server side is considered a resource, which developers can access in a uniform way using web URIs (Uniform Resource Identifiers) and HTTP. Because REST uses HTTP as the communication protocol, the REST style is constrained to a stateless client/server architecture.

Jersey JAX-RS Web Server

- For Java developers, JAX-RS (JSR 311) provides an API for creating RESTful web services in Java
- The JAX-RS API uses annotations to simplify the development of RESTful web services
- As with any other Java web application, you bundle JAX-RS applications as a WAR file and deploy them on a container that supports Servlets (For example a Tomcat server or a Glassfish server).
- Sun offers a reference implementation for JAX-RS code-named Jersey. Jersey uses a HTTP web server called Grizzly, and the Servlet Grizzly Servlet (com.sun.jersey.spi.container.servlet.ServletContainer) handles the requests to Grizzly

Jersey JAX-RS Web Server

- Resource methods are public methods of a resource class that you identify with a request method designator. Request method designators are annotations that you use to identify the methods that handle the HTTP requests, and JAX-RS defines annotations for HTTP methods such as GET, POST, PUT, DELETE, and HEAD. JAX-RS also allows you to create user-defined custom request method designators.
- JAX-RS provides a clear mapping between the HTTP protocol and the URIs through well-defined classes and interfaces

Links:

1. Google Protocol Buffers Java documentation:

http://code.google.com/intl/ro-RO/apis/protocolbuffers/docs/javatutorial.html

2. Jersey documentation:

http://jersey.java.net/nonav/documentation/latest/user-guide.html#d4e1972

3. Simple application using Jersey and Google Protocol Buffers:

http://www.javarants.com/2008/12/27/using-jax-rs-with-protocol-buffers-for-high-performance-rest-apis/

4. Using Internet data in Android applications:

http://www.ibm.com/developerworks/opensource/library/x-dataAndroid/index.html

5. JAX-RS: Developing RESTful Web Services in Java:

http://www.devx.com/Java/Article/42873/1954

6. Maven (for building Java projects, useful for installation/deploying the application):

http://maven.apache.org/guides/getting-started/index.html#How_do_I_compile_my_application_sources