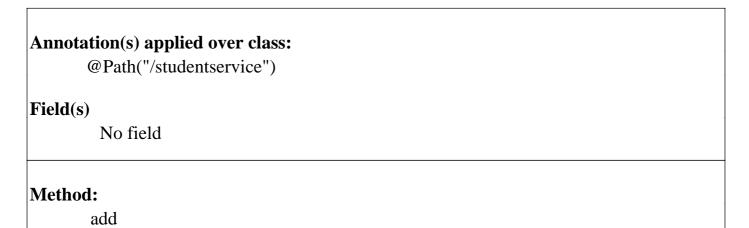# List of Service classes

bobby.services.AutoWiredTest

Annotation(s) applied over class:
    @Path("/autowire")
    @GET

Field(s)
    class bobby.pojo.Student student   @AutoWired(name="xyz")

Method:
    invoke

Annotation(s) applied over method:
    @Path("/invoke")

Parameters:
    Zero parameters

Return Type:
    void

bobby.services.StudentService

Annotation(s) applied over class:
    @Path("/studentservice")

Field(s)
    No field

Method:
    add

Annotation(s) applied over method:

@Path("/add")
@Forward("/studentservice/getall")
@POST

Parameters:

class bobby.pojo.Student
class com.thinking.machines.webrock.pojo.ApplicationScope

Return Type:

void

Method:

delete

Annotation(s) applied over method:

@Path("/delete")
@GET

Parameters:

int

Return Type:

void

Method:

getAll

Annotation(s) applied over method:

@Path("/getall")
@GET

Parameters:

Zero parameters

Return Type:

interface java.util.List

Method:

getByStudentRoll

Annotation(s) applied over method:

@Path("/getby")
@GET

Parameters:
    int

Return Type:
    class bobby.pojo.Student

Method:
    edit

Annotation(s) applied over method:
    @Path("/edit")
    @POST

Parameters:
    class bobby.pojo.Student

Return Type:
    void

---

bobby.services.TeacherService

Annotation(s) applied over class:
    @Path("/teacherservice")

Field(s)
    int t

Method:
    add

Annotation(s) applied over method:
    @Path("/add")
    @Forward("/teacherservice/getall")
    @POST

Parameters:
    class bobby.pojo.Teacher

Return Type:

void

Method:
     delete

Annotation(s) applied over method:
     @Path("/delete")
     @GET

Parameters:
     int

Return Type:
     void

Method:
     edit

Annotation(s) applied over method:
     @Path("/edit")
     @POST

Parameters:
     class bobby.pojo.Teacher

Return Type:
     void

Method:
     getByTeacherRoll

Annotation(s) applied over method:
     @Path("/getby")
     @GET

Parameters:
     int
     class java.lang.String
     class com.thinking.machines.webrock.pojo.ApplicationScope

Return Type:
     class bobby.pojo.Teacher

bobby.test.applicationdirectorytest.Peon

Annotation(s) applied over class:
> @Path("/peon")
> @GET
> @InjectApplicationDirectory

Field(s)
> class com.thinking.machines.webrock.pojo.ApplicationDirectory applicationDirectory

Method:
> get

Annotation(s) applied over method:
> @Path("/get")

Parameters:
> class com.thinking.machines.webrock.pojo.ApplicationDirectory

Return Type:
> int

bobby.test.autowiredtest.AutoWiredTest

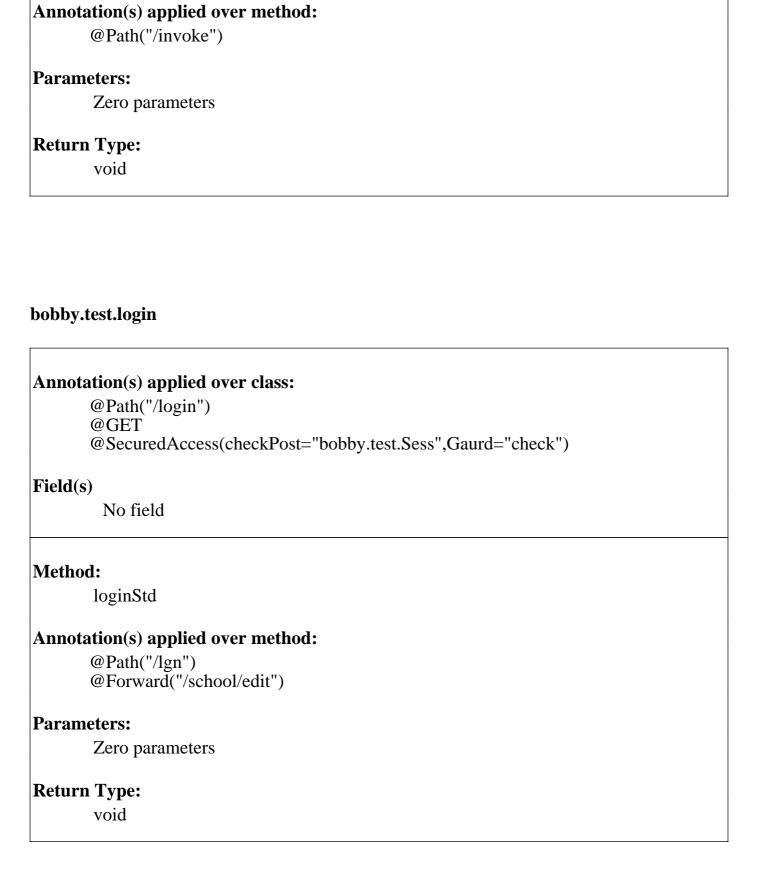**Annotation(s) applied over class:**
> @Path("/autowire")
> @GET

Field(s)
> class bobby.pojo.Student student   @AutoWired(name="xyz")

Method:
> invoke

Annotation(s) applied over method:
    @Path("/invoke")

Parameters:
    Zero parameters

Return Type:
    void

bobby.test.login

Annotation(s) applied over class:
    @Path("/login")
    @GET
    @SecuredAccess(checkPost="bobby.test.Sess",Gaurd="check")

Field(s)
    No field

Method:
    loginStd

Annotation(s) applied over method:
    @Path("/lgn")
    @Forward("/school/edit")

Parameters:
    Zero parameters

Return Type:
    void

bobby.test.requesttest.Account

Annotation(s) applied over class:
  @Path("/account")
  @GET
  @InjectRequestScope

Field(s)
  class com.thinking.machines.webrock.pojo.RequestScope requestScope

Method:
  add

Annotation(s) applied over method:
  @Path("/add")
  @Forward("/forwardTest.html")

Parameters:
  Zero parameters

Return Type:
  void

Method:
  get

Annotation(s) applied over method:
  @Path("/get")

Parameters:
  Zero parameters

Return Type:
  int

bobby.test.Sess

Annotation(s) applied over class:
  @Path("/sess")
  @InjectSessionScope

Field(s)

class com.thinking.machines.webrock.pojo.SessionScope sessionScope

bobby.test.sessiontest.School

Annotation(s) applied over class:
    @Path("/school")
    @GET
    @InjectSessionScope

Field(s)
    class com.thinking.machines.webrock.pojo.SessionScope sessionScope

Method:
    get

Annotation(s) applied over method:
    @Path("/get")

Parameters:
    Zero parameters

Return Type:
    int

Method:
    edit

Annotation(s) applied over method:
    @Path("/edit")
    @Forward("/welcome.html")

Parameters:
    Zero parameters

Return Type:
    void

List of ServiceException

1) ServicePackagePrefixException:

No such package exists, which contains the value mentioned against SERVICE_PACKAGE_PREFIX in web.xml as a prefix.

2) RequestTypeConflictException:

You could not use both @GET and @Post annotations at a time on a single method or class.

3) LeaningOnPathException:

You did not implement the @Path annotation with one of the following annotations @GET, @ POST, @Forward ,@InjectSessionScope ,@InjectRequestScope ,@InjectApplicationScope ,@InjectApplicationDirectoryScope ,@SecuredAccess

4) LeaningOnRequestTypeException:

You did not implement the @GET or @POST annotation with one of the following annotations @Forward ,@SecuredAccess.

5) OnStartupException:

You cannot implement @OnStartup annotation along with other annotations present in the WebServicesFramework.

6) IllegalInjectRequestParameterException:

You cannot apply @InjectRequestParameter annotation with data types other than primitive and java.lang.String.

7) ExcessParameterTypeException:

You cannot use more than one complex data type {except : java.lang.String,ApplicationScope,RequestScope,SessionScope,ApplicationDirectory} in single function over which @Path annotation is applied.

8) ClassInstantiationException:

The instantiation can fail for a variety of reasons including but not limited to:
(i) the class object represents an abstract class, an interface, an array class, a primitive type, or void.
(ii) the class has no nullary constructor.

9) InvalidURLRequestException:

This exception occurs when you request some URL and it is not found.

10) IllegalMethodException:

This exception occurs when you try to access resources through a method that is not allowed on that resource.

11) IllegalGaurdParameterException:

This exception occurs when your gaurd method has parameters other than {ApplicationScope, SessionScope, RequestScope, ApplicationDirectory}.

12) IllegalServiceAccessException:

An IllegalAccessException is thrown when an application tries to reflectively create an instance (other than an array), set or get a field, or invoke a method, but the currently executing method does not have access to the definition of the specified class, field, method or constructor.

13) InvalidNumberOfParameterException:

Method to which request has been forwarded has either 0 parameter or more than 1 parameters.