Siminov Framework Getting Started Guide

Siminov Framework
http://www.siminov.github.io/android-hybrid

Contents

Preface

A hybrid application, by definition is derived from a combination of technologies, approaches or elements of different kinds. With respect to mobile applications, a hybrid application leverages best of both native and mobile web technologies.

In hybrid environment, it is very difficult to map JavaScript/Java objects to relational database, but Siminov makes application developer life easy and simiple by mapping JavaScript/Java objects to relational database.

> **Note**
> While having a strong background in SQL is not required to use Android-Siminov Framework, having a basic understanding of the concepts can greatly help you understand Siminov more fully and quickly. Probably the single best background is an understanding of data modeling principles. You might want to consider these resources as a good starting point: `http://en.wikipedia.org/wiki/Data_modeling`

Siminov not only takes care of the mapping from JavsScript/Java classes to database tables (and from JavaScript/Java data types to SQL data types), but also provides data query and retrieval facilities. It can significantly reduce development time otherwise spent with manual data handling in SQLite. Siminov design goal is to relieve the developer from 99% of common data persistence-related programming tasks by eliminating the need for manual, hand-crafted data processing using SQLite. However, unlike many other persistence solutions, Siminov does not hide the power of SQLite from you and guarantees that your investment in relational technology and knowledge is as valid as always.

Siminov may not be the best solution for data-centric applications that only use stored-procedures to implement the business logic in the database, it is most useful with Hybrid object-oriented domain models and business logic in the JavaScript/Java based. However, Siminov can certainly help you to remove or encapsulate vendor-specific SQLite code and will help with the common task of result set translation from a tabular representation to a graph of objects.

0.1 Get Involved

Use Siminov and report any bugs or issues you find. Try your hand at fixing some bugs or implementing enhancements. Engage with the community using mailing lists, forums, IRC, or other ways listed at `https://github.com/Siminov`. Help improve or translate this documentation. Contact us on the developer mailing list if you have interest. Spread the word. Let the rest of your organization know about the benefits of Siminov Framework.

Siminov Native ORM

1. **Web Site**: `https://siminov.github.io.com/android-orm`
2. **Github Website**: `https://github.com/Siminov/android-orm`
3. **Wiki**: `https://github.com/Siminov/android-orm/wiki`
4. **Issue Tracker**: `https://github.com/Siminov/android-orm/issues`

Siminov Hybrid ORM Framework

1. **Web Site**: `https://siminov.github.io.com/android-hybrid`
2. **Github Website**: `https://github.com/Siminov/android-hybrid`
3. **Wiki**: `https://github.com/Siminov/android-hybrid/wiki`
4. **Issue Tracker**: `https://github.com/Siminov/android-hybrid/issues`

0.2 Getting Started Guide

New users may want to first look through the Siminov Getting Started Guide for basic information as well as tutorials. Even seasoned veterans may want to considering perusing the sections pertaining to build artifacts for any changes.

Chapter 1

About Siminov

Siminov is a open source framework build and managed by Siminov Software Solution LLP and its community. It is a free software that is distributed under the Apache License, Version 2.0 (the "License"). Aim of this framework is to reduce time and effort spend in building mobile application on various platforms like (Android, iOS, Blackberry, Windows).

Siminov Framework is build by combining various Siminov products like (Siminov ORM , Siminov Hybrid, Siminov Konnect). These components are available or in development stage on all platforms.

1.1 Siminov Android ORM

Siminov Android ORM is an object-relational mapping (ORM) library for the Native Android Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. It primary maps Android Java classes to database tables (and from Java data types to SQLite data types.). It also provides data query and retrieval facilities. It also generates the SQLite calls and attempts to relieve the developer from manual result set handling and object conversion and keep the application protable to all supported SQLite databases with little performance overhead.

1.2 Siminvo Hybrid

Siminov Hybrid is build using Siminov Android ORM Framework and Siminov JavaScript Framework, together it enables application to map JavaScript/Java object to relational database. Using this application developer can build application which uses the best of both Native and Web technology.

Siminov Hybrid not only take care of mapping from JavaScript/Java classes to database tables (and from JavaScript/Java data types to SQL data types), but also provides data query and retrieval facilities.

Chapter 2

Obtaining Siminov

2.0.1 Release Bundle Downloads

The Siminov Framework team provide release bundles hosted at **Github** File Release System, in ZIP and TGZ formats. Each release bundle contains JARs, documentation, source code, and other goodness.

You can download releases of Siminov Framework, in your chosen format, from the list at `http://siminov.github.io/android-hybrid/builds.html`.

1. The android-orm/lib/directory contan all the JARs a Siminov Android ORM requires.

2. The android-hybrid/lib/directory contain all the JARs a Siminov Hybrid ORM requires.

Note
If you want to build only native based application then you need only Siminov Native ORM Framework no need to include Siminov Hybrid ORM Framework in your class path.

Chapter 3

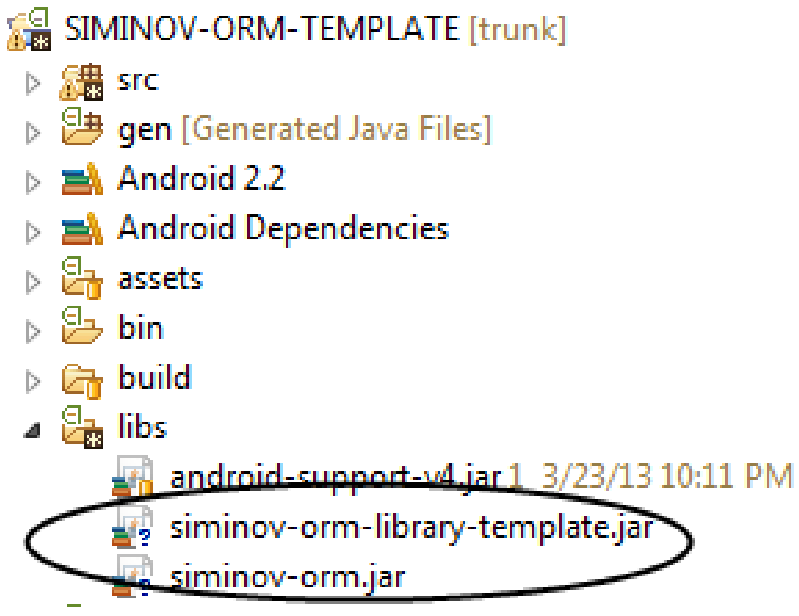
Setting Up Application

3.1 Siminov Android ORM

1. Download Siminov Android ORM jar from `http://siminov.github.io/android-hybrid/builds.html`

2. Add native orm jar into your application libs folder.

   Example: Siminov ORM Template Application

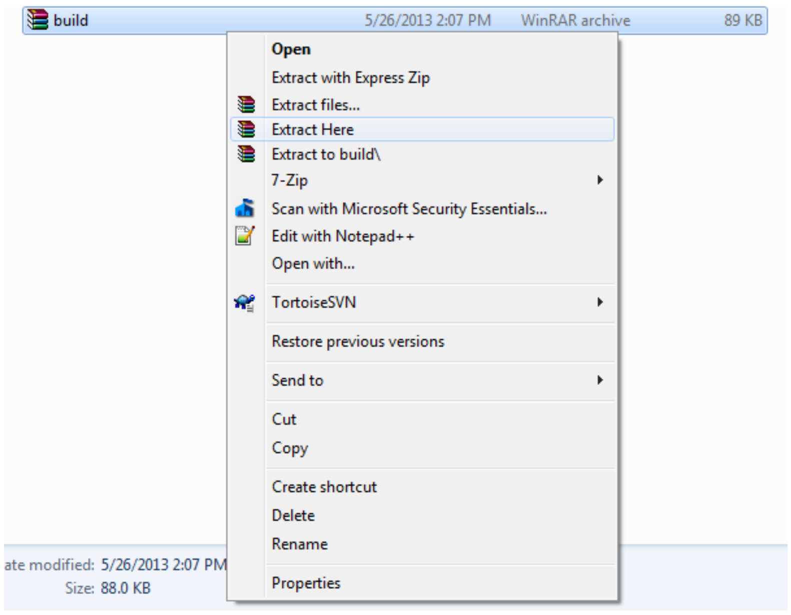


SIMINOV-ORM-TEMPLATE [trunk]
- src
- gen [Generated Java Files]
- Android 2.2
- Android Dependencies
- assets
- bin
- build
- libs
  - android-support-v4.jar 1 3/23/13 10:11 PM
  - siminov-orm-library-template.jar
  - siminov-orm.jar

3.2 Siminov Hybrid

1. Download Siminov Android ORM jar from `http://siminov.github.io/android-hybrid/builds.html`
2. Download Siminov Hybrid build from `http://siminov.github.io/android-hybrid/builds.html`
3. Extract Siminov Hybrid build into folder.

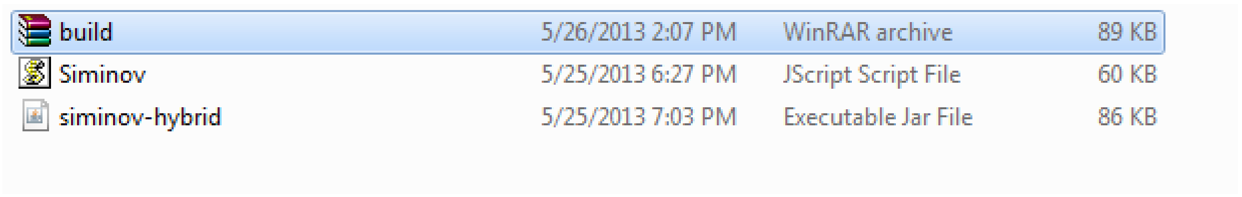
   Example: Extract Siminov Build



build 5/26/2013 2:07 PM WinRAR archive 89 KB

ate modified: 5/26/2013 2:07 PM
Size: 88.0 KB

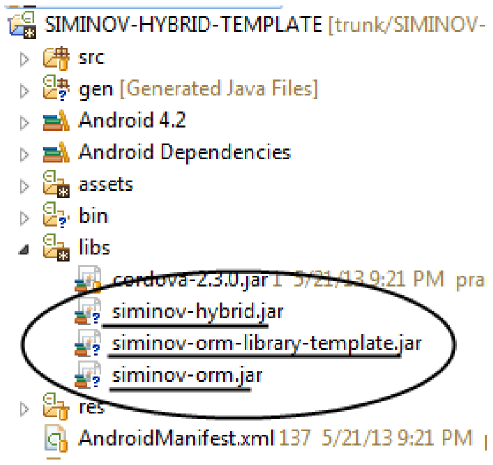
   Example: After Extracting Siminov Hybrid Build



build	5/26/2013 2:07 PM	WinRAR archive	89 KB
Siminov	5/25/2013 6:27 PM	JScript Script File	60 KB
siminov-hybrid	5/25/2013 7:03 PM	Executable Jar File	86 KB

4. Add both native and hybrid jar's into your application libs folder.

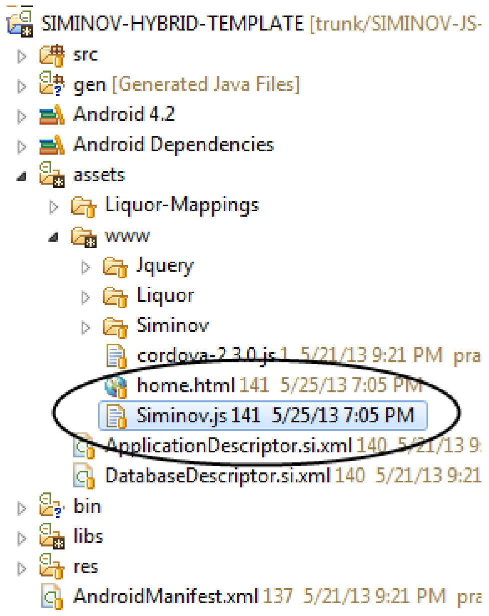
   Example: Siminov Hybrid Template Application



```
SIMINOV-HYBRID-TEMPLATE [trunk/SIMINOV-
  src
  gen [Generated Java Files]
  Android 4.2
  Android Dependencies
  assets
  bin
  libs
    cordova-2.3.0.jar 1  5/21/13 9:21 PM  pra
    siminov-hybrid.jar
    siminov-orm-library-template.jar
    siminov-orm.jar
  res
  AndroidManifest.xml 137  5/21/13 9:21 PM
```

5. Add Siminov JavaScript (Siminov.js) to your assets folder.

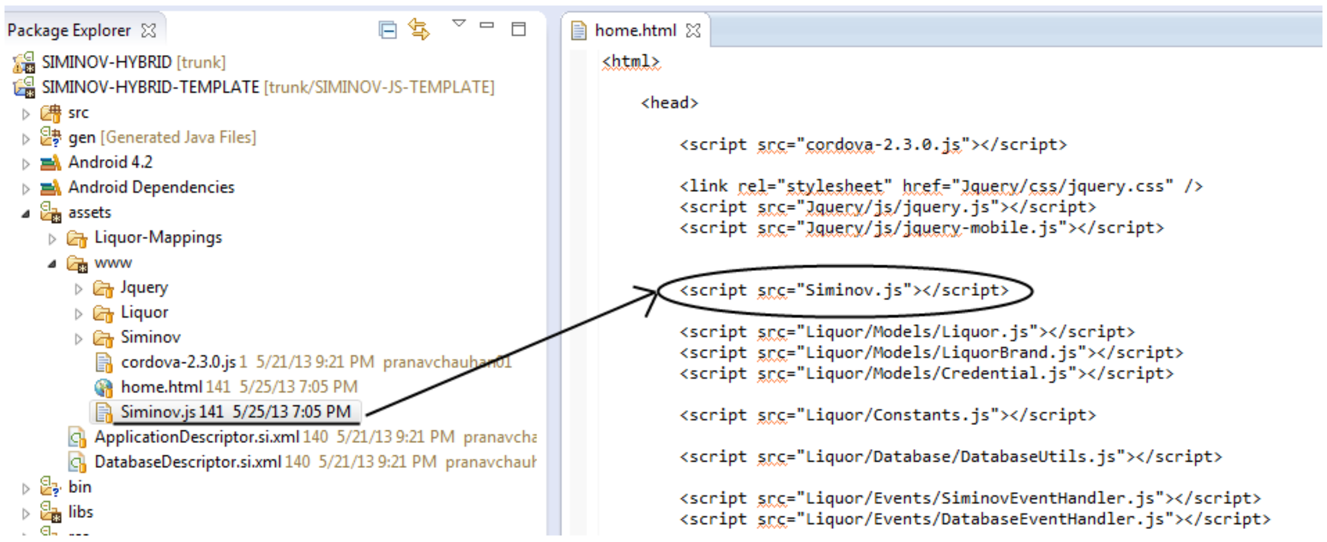
   Example: Siminov Hybrid Template Application



```
SIMINOV-HYBRID-TEMPLATE [trunk/SIMINOV-JS-
  src
  gen [Generated Java Files]
  Android 4.2
  Android Dependencies
  assets
    Liquor-Mappings
    www
      Jquery
      Liquor
      Siminov
      cordova-2.3.0.js 1  5/21/13 9:21 PM  pra
      home.html 141  5/25/13 7:05 PM
      Siminov.js 141  5/25/13 7:05 PM
    ApplicationDescriptor.si.xml 140  5/21/13 9:
    DatabaseDescriptor.si.xml 140  5/21/13 9:21
  bin
  libs
  res
  AndroidManifest.xml 137  5/21/13 9:21 PM  pra
```

6. Include Siminov JavaScript (Siminov.js) in your view.

   Example: Siminov Hybrid Template Application

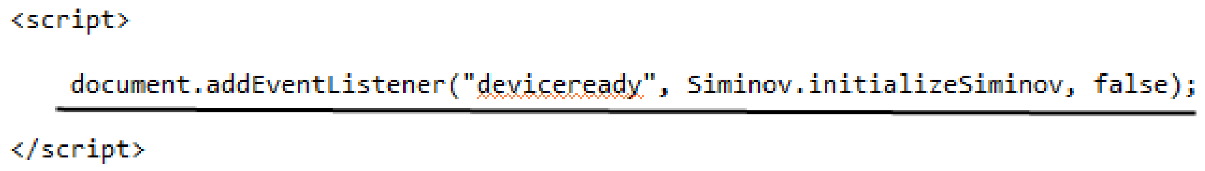


```
SIMINOV-HYBRID [trunk]
SIMINOV-HYBRID-TEMPLATE [trunk/SIMINOV-JS-TEMPLATE]
  src
  gen [Generated Java Files]
  Android 4.2
  Android Dependencies
  assets
    Liquor-Mappings
    www
      Jquery
      Liquor
      Siminov
      cordova-2.3.0.js 1  5/21/13 9:21 PM  pranavchauhan01
      home.html 141  5/25/13 7:05 PM
      Siminov.js 141  5/25/13 7:05 PM
    ApplicationDescriptor.si.xml 140  5/21/13 9:21 PM  pranavcha
    DatabaseDescriptor.si.xml 140  5/21/13 9:21 PM  pranavchauh
  bin
  libs
```

```
<html>

    <head>

        <script src="cordova-2.3.0.js"></script>

        <link rel="stylesheet" href="Jquery/css/jquery.css" />
        <script src="Jquery/js/jquery.js"></script>
        <script src="Jquery/js/jquery-mobile.js"></script>

        <script src="Siminov.js"></script>

        <script src="Liquor/Models/Liquor.js"></script>
        <script src="Liquor/Models/LiquorBrand.js"></script>
        <script src="Liquor/Models/Credential.js"></script>

        <script src="Liquor/Constants.js"></script>

        <script src="Liquor/Database/DatabaseUtils.js"></script>

        <script src="Liquor/Events/SiminovEventHandler.js"></script>
        <script src="Liquor/Events/DatabaseEventHandler.js"></script>
```

7. Invoke Siminov Initialization on device ready event.

   Example: Siminov Hybrid Template Application



```
<script>

    document.addEventListener("deviceready", Siminov.initializeSiminov, false);

</script>
```

Chapter 4

Building Application

Building application based on Siminov Android ORM/Siminov Hybrid Framework is easy and simple. By the help of **Siminov Hybrid Template Application** we will explain how we can build application.

4.1 Configure ApplicationDescriptor.si.xml

Application Descriptor is the one who connects application to Siminov Framework. It provide basic information about application, which defines the behaviour of application.

```
<siminov>

  <property name="name">SIMINOV HYBRID TEMPLATE</property>
  <property name="description">Siminov Hybrid Template Application</
      property>
  <property name="version">0.9</property>

  <property name="load_initially">true</property>

  <!-- DATABASE-DESCRIPTORS -->
  <database-descriptors>
    <database-descriptor>DatabaseDescriptor.si.xml</database-
        descriptor>
  </database-descriptors>


  <!-- SIMINOV EVENTS -->
  <event-handlers>
      <event-handler>siminov.hybrid.template.events.
          SiminovEventHandler/SiminovEventHandler</event-handler>
      <event-handler>siminov.hybrid.template.events.
          DatabaseEventHandler/DatabaseEventHandler</event-handler>
  </event-handlers>

</siminov>
```

4.2 Configure DatabaseDescriptor.si.xml

Database Descriptor is the one who defines the schema of database.

```

<database-descriptor>

  <property name="database_name">SIMINOV-HYBRID-TEMPLATE</property>
  <property name="description">Siminov Hybrid Template Database
      Config</property>
  <property name="is_locking_required">true</property>
  <property name="external_storage">false</property>

  <database-mappings>
    <database-mapping path="Liquor-Mappings/Liquor.si.xml" />
    <database-mapping path="Liquor-Mappings/LiquorBrand.si.xml" />
  </database-mappings>

  <libraries>
    <library>siminov.orm.library.template.resources</library>
  </libraries>

</database-descriptor>
```

4.3 Configure LibraryDescriptor.si.xml

Library Descriptor is the one who de

nes the properties of library.

```
<library>

  <property name="name">SIMINOV LIBRARY TEMPLATE</property>
  <property name="description">Siminov Library Template</property>
  <!-- Database Mappings -->
  <database-mappings>
    <database-mapping path="Credential.si.xml" />
  </database-mappings>

</library>
```

4.4 Configure DatabaseMappingDescriptor.si.xml

Database Mapping Descriptor is one which does ORM, it maps POJO class to database table.

```
<database-mapping>

  <table table_name="LIQUOR" class_name="siminov.hybrid.template.
      model.Liquor/Liquor">

    <column variable_name="liquorType" column_name="LIQUOR_TYPE">
      <property name="type">java.lang.String/String</property>
      <property name="primary_key">true</property>
      <property name="not_null">true</property>
      <property name="unique">true</property>
    </column>

    <column variable_name="description" column_name="DESCRIPTION">
      <property name="type">java.lang.String/String</property>
    </column>

    <column variable_name="history" column_name="HISTORY">
      <property name="type">java.lang.String/String</property>
    </column>

    <column variable_name="link" column_name="LINK">
      <property name="type">java.lang.String/String</property>
      <property name="default">www.wikipedia.org</property>
    </column>

    <column variable_name="alcholContent" column_name="ALCHOL_CONTENT
        ">
      <property name="type">java.lang.String/String</property>
    </column>

    <index name="LIQUOR_INDEX_BASED_ON_LINK" unique="true">
```

```
      <column>HISTORY</column>
    </index>

    <relationships>

        <one-to-many refer="liquorBrands" refer_to="siminov.hybrid.
            template.model.LiquorBrand/LiquorBrand" on_update="cascade
            " on_delete="cascade">
        <property name="load">true</property>
      </one-to-many>

    </relationships>

  </table>

</database-mapping>
```

4.5 Using Annotation

Liquor pojo class of Siminov template application.

```
@Table(tableName=Liquor.TABLE_NAME)
@Indexes({
  @Index(name="LIQUOR_INDEX_BASED_ON_LINK", unique=true, value={
    @IndexColumn(column=Liquor.LINK)
  }),
})
public class Liquor extends Database implements Serializable {

  //Table Name
  transient public static final String TABLE_NAME = "LIQUOR";

  //Column Names
  transient public static final String LIQUOR_TYPE = "LIQUOR_TYPE";
  transient public static final String DESCRIPTION = "DESCRIPTION";
  transient public static final String HISTORY = "HISTORY";
  transient public static final String LINK = "LINK";
  transient public static final String ALCHOL_CONTENT = "
      ALCHOL_CONTENT";

  //Liquor Types
  transient public static final String LIQUOR_TYPE_GIN = "Gin";
  transient public static final String LIQUOR_TYPE_RUM = "Rum";
  transient public static final String LIQUOR_TYPE_TEQUILA = "Tequila
      ";
```

```
transient public static final String LIQUOR_TYPE_VODKA = "Vodka";
transient public static final String LIQUOR_TYPE_WHISKEY = "Whiskey
   ";
transient public static final String LIQUOR_TYPE_BEER = "Beer";
transient public static final String LIQUOR_TYPE_WINE = "Wine";


//Variables

@Column(columnName=LIQUOR_TYPE,
     properties={
       @ColumnProperty(name=ColumnProperty.PRIMARY_KEY, value="true"
           ),
       @ColumnProperty(name=ColumnProperty.NOT_NULL, value="true"),
       @ColumnProperty(name=ColumnProperty.UNIQUE, value="true")
       })
private String liquorType = null;

@Column(columnName=DESCRIPTION)
private String description = null;

@Column(columnName=HISTORY)
private String history = null;

@Column(columnName=LINK,
     properties={
       @ColumnProperty(name=ColumnProperty.DEFAULT, value="www.
           wikipedia.org")
       })
private String link = null;

@Column(columnName=ALCHOL_CONTENT)
private String alcholContent = null;

@OneToMany(onUpdate="cascade", onDelete="cascade",
     properties={
       @RelationshipProperty(name=RelationshipProperty.LOAD, value="
           true")
  })
private ArrayList<LiquorBrand> liquorBrands = null;

//Methods

public String getLiquorType() {
  return this.liquorType;
}

public void setLiquorType(String liquorType) {
  this.liquorType = liquorType;
```

```java
  }

  public String getDescription() {
    return this.description;
  }

  public void setDescription(String description) {
    this.description = description;
  }

  public String getHistory() {
    return this.history;
  }

  public void setHistory(String history) {
    this.history = history;
  }

  public String getLink() {
    return this.link;
  }

  public void setLink(String link) {
    this.link = link;
  }

  public String getAlcholContent() {
    return this.alcholContent;
  }

  public void setAlcholContent(String alcholContent) {
    this.alcholContent = alcholContent;
  }

  public ArrayList<LiquorBrand> getLiquorBrands() {
    return this.liquorBrands;
  }

  public void setLiquorBrands(ArrayList<LiquorBrand> liquorBrands) {
    this.liquorBrands = liquorBrands;
  }
}
```

4.6 Saving Entity Object

Saving liquor entity object.

1. **Java**

```
Liquor beer = new Liquor();
beer.setLiquorType("Beer");
beer.setDescription("Beer Description");
beer.setHistory("Beer History");
beer.setLink("Beer Web Link");
beer.setAlcholContent("Beer Alchol Content");

try {
  beer.save();
} catch(DatabaseException databaseException) {
  //Log It.
}
```

2. **JavaScript**

```
var beer = new Liquor();
beer.setLiquorType("Beer");
beer.setDescription("Beer Description");
beer.setHistory("Beer History");
beer.setLink("Beer Web Link");
beer.setAlcholContent("Beer Alchol Content");

try {
  beer.save();
} catch(DatabaseException databaseException) {
  //Log It.
}
```

4.7 Updaing Entity Object

Updating liquor object.

1. **Java**

```
    Liquor beer = new Liquor();
    beer.setLiquorType("Beer");
    beer.setDescription("Beer Description");
    beer.setHistory("Beer History");
    beer.setLink("Beer Web Link");
    beer.setAlcholContent("Beer Alchol Content");

    try {
      beer.update();
    } catch(DatabaseException databaseException) {
      //Log It.
    }
```

2. **JavaScript**

```
    var beer = new Liquor();
    beer.setLiquorType("Beer");
    beer.setDescription("Beer Description");
    beer.setHistory("Beer History");
    beer.setLink("Beer Web Link");
    beer.setAlcholContent("Beer Alchol Content");

    try {
      beer.update();
    } catch(DatabaseException databaseException) {
      //Log It.
    }
```

4.8 Save Or Update Entity Object

Save or update liquor object.

1. **Java**

```
    Liquor beer = new Liquor();
    beer.setLiquorType("Beer");
    beer.setDescription("Beer Description");
    beer.setHistory("Beer History");
    beer.setLink("Beer Web Link");
    beer.setAlcholContent("Beer Alchol Content");

    try {
```

```
    beer.saveOrUpdate();
  } catch(DatabaseException databaseException) {
    //Log It.
  }
```

2. **JavaScript**

```
  var beer = new Liquor();
  beer.setLiquorType("Beer");
  beer.setDescription("Beer Description");
  beer.setHistory("Beer History");
  beer.setLink("Beer Web Link");
  beer.setAlcholContent("Beer Alchol Content");

  try {
    beer.saveOrUpdate();
  } catch(DatabaseException databaseException) {
    //Log It.
  }
```

4.9 Deleting Entity Object

Deleting liquor object.

1. **Java**

```
  Liquor beer = new Liquor();
  beer.setLiquorType("Beer");
  beer.setDescription("Beer Description");
  beer.setHistory("Beer History");
  beer.setLink("Beer Web Link");
  beer.setAlcholContent("Beer Alchol Content");

  try {
    beer.delete().execute();
  } catch(DatabaseException databaseException) {
    //Log It.
  }
```

2. **JavaScript**

```
var beer = new Liquor();
beer.setLiquorType("Beer");
beer.setDescription("Beer Description");
beer.setHistory("Beer History");
beer.setLink("Beer Web Link");
beer.setAlcholContent("Beer Alchol Content");

try {
  beer.delete().execute();
} catch(DatabaseException databaseException) {
  //Log It.
}
```