The Brijframework is a lightweight and open source framework. It was developed by Ram Kishor in 2015. Brijframework makes the easy development of Java EE application. It provides support to various frameworks. Can be defined as a structure where we find solution of the various technical problems.

The Brijframework comprises several modules such as Model, Bean, Context, JDBC, DAO, ORM, WEB MVC, WEB REST etc.

**[WHY BRIJFRAMEWORK?](file:///E:\\brijframework-project\\brijframework\\brijframework-docs\\src\\main\\webapp\\docs.html)**

Java is great for declaring object, but it falters when we try to use it for declaring dynamic manipulation properties in object. Brijframework lets you extend Java vocabulary for your application. The resulting environment is extraordinarily expressive, readable, and quick to develop.

**[ALTERNATIVES](file:///E:\\brijframework-project\\brijframework\\brijframework-docs\\src\\main\\webapp\\docs.html)**

Many frameworks deal with Java shortcomings by either abstracting away Java by providing an imperative way for manipulating the Object. Neither of these address the root problem that Java was not designed for dynamic object manipulating.

**[EXTENSIBILITY](file:///E:\\brijframework-project\\brijframework\\brijframework-docs\\src\\main\\webapp\\docs.html)**

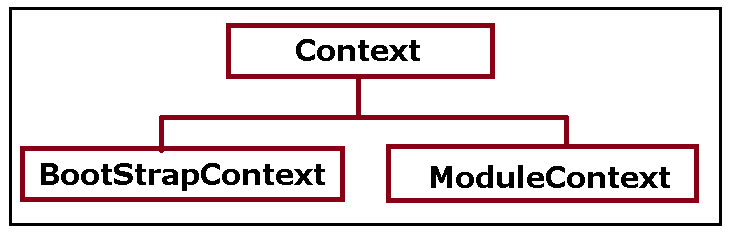
Brijframework is a toolset for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs. Read on to find out how.

**Context**

Context is the mechanism of framework which is launch the application and there module. It’s responsible start and stop all the relative container.

The interface provides a common protocol for launching container. It provides methods to start and stop in an orderly manner.

There are two type of context.



Bootstrap Context

Bootstrap Context responsible to launch all related module context at the application level.

Basic module context are there-

Example:

Application Context

**Module Context**

Module Context responsible to launch all related container at the application level. Basic module context are there-

Example:

Environment Context, Logger Context, Resource Context, Modal Context, Bean Context

**Container**

Container is a mechanism to cache component in globally for application. It’s responsible to launch all related factories.

The interface provides a common protocol for launching the factories. It provides methods to load and build in an orderly manner.

## Getting Started With Model: Your First App

#### Model

Model is a mechanism for accessing an object's properties indirectly, using keys to identify properties, rather than through invocation of an accessor method or accessing them directly through instance variables.

The interface provides a common protocol for all objects for implementing key value coding. It provides methods to set and get property values in an orderly manner without getter setter methods of the class explicitly

No Requirement of setter/ getter to access the property of objects

**public class** Address **implements** ModelBean{

**private** String addressLine;

**private** String city;

**private** long zipcode;

// no setter getter

}

No Requirement of setter/ getter to access the relative property of objects

**public class**  Employee **implements** ModelBean{

**private** String id;

**private** String name;

**private long** rollNo;

**private** Address address;

// no setter getter

}