# Table of Contents

A life cycle object is basically a container for any objects and methods/actions that are administered on the server side. Life cycle objects are used to provide business logic to the client.

Each life cycle object has a predefined life cycle. When the cycle ends, the object is discarded along with all other objects it administers. This process optimizes the application server´s memory usage.

The following graph gives an overview of the life cycles within an application:

The example shows two different applications (Application-A and Application-B), which successively access the server´s business logic.

Each application requires a MasterConnection to access the server. Using the MasterConnection the business logic can be accessed using calls.

The respective life cycle object is instantiated at the first call of a connection, so the call can be assigned to the correct business object. The life cycle object remains in place until the respective connection is closed. At this time all administered objects are discarded and the used memory is freed.

SubConnections behave the same way as MasterConnections.

The applications business object is an exception. It is a life cycle object that is instantiated at the first call of an application and that remains in place until the application server is stopped. Only one applications life cycle object exists per application, regardless of how many users are registered with the application.

## Note

A life cycle object should be used stateless, as the server should be able to instantiate life cycle objects on demand. This means that all saved information/objects are discarded!

This behavior has to be considered for user-defined objects, but does not mean that stateful objects are not supported.