




ZIM 9.10

Basic ZOM



What is Zim?

Zim is



a complete framework to develop and run professional and mission critical applications by tightly integrating a lean relational database, a powerful Fourth Generation Language, an integrated development tool, the integration with outside world and client user interfaces.



ZOM

The Zim Object Management (ZOM) services provides extensive support for managing the objects that make up Zim applications.

ZOM is a collection of services that assists the developer in manipulating, analyzing, and maintaining application objects, including programs, in a fast and secure way.

For the most part, it works behind the scenes. This tutorial, then, only presents the most common functionalities not requiring extensive knowledge of ZOM.

These services also support you in managing development projects, migrating applications to new computer systems, and building custom development tools such as program generators.



Enabling ZOM



The first step to use ZOM, either right after creating a new Zim database or upon starting a new ZimQTC session, is to enable ZOM.

`ZOMEnable`

This function initializes ZOM for the session, performs a database sanity check and searches for updates.



Listing Objects

% The option +n means "include this name" and is optional.

```
ZOMList +n fMain
```

```
ZOMList fMain
```

% These two commands are the same.

% Lists all objects having "main" in the name.

```
ZOMList *main
```

% Lists all Forms and Displays.

```
ZOMList +t forms,displays
```

% Lists all objects but Constants and Windows.

```
ZOMList * -t Constants, Windows
```

% Lists all Forms but fMain.

```
ZOMList +t Forms -n fMain
```



Exporting Objects

One of the most useful ZOM functions is the ability to ZOMExport ZIM objects using exactly the same selection criteria syntax shown in ZOMList.

It can be used to save the DD definitions, migrate to another platform, maintain Zim databases up to date, migrate from older versions Zim to newer, etc.

% Exports all objects.

ZOMExport *

% Exports only Forms and Windows.

ZOMExport +t Forms, Windows

ZOMExport writes the selected definitions to physical files in the PORT directory located in the current Zim database.



Importing Objects

After ZOMExporting the desired objects and copying the entire PORT directory from the origin database to the destination database, already in the destination database, the ZOMImport is used to import all definitions.

**% Imports all objects previously exported.
ZOMImport**

ZOMImport will compare the about to be imported objects against its own object definitions and then import only those that have changed.

From this point on, at least in respect to those objects originally exported, both Zim databases are synchronized.



Saving Data

For Zim objects with data like Entity Sets and Relationships with Fields, the commands ZOMDataSave and ZOMDataLoad can be used to save existing data to text files and then load them back to its proper place.

% Saves all data.

ZOMDataSave *

% Saves data from the Entity Set Customers.

ZOMDataSave Customers

Saved data is written to text files in the directory ZOMSaving located in the current database.

Attention: ZOMDataSave and ZOMDataLoad are not Unicode safe. Use the commands DataSave and DataLoad instead.



Loading Data

After ZOMDataSaving the desired objects and copying the entire ZOMSaving directory from the origin database to the destination database, already in the destination database, the ZOMDataLoad is used to reload the saved data.

`% Loads all saved data.`

`ZOMDataLoad *`

`% Loads data from the Entity Set Customers.`

`ZOMDataLoad Customers`

Attention: in some Zim versions, data will not be loaded as expected unless a “cold” ZOMDataSave is performed before the load process.



Copying an Object



The command ZOMCopy replicates a specific Zim object onto another Zim object.

```
ZOMCopy object > anotherobject
```

The single object **object** is duplicated with the name **anotherobject**.



Destroying Objects

When Zim objects are no longer needed, they can be destroyed by the command ZOMDestroy.



% Destroys fMain, wMain, mMain and dMain.

```
ZOMDestroy *Main
```

% Destroys all Forms but fMain.

```
ZOMDestroy +t Forms -n fMain
```

ZimIDE is also capable of deleting (destroying) objects but it only does it one at a time by right-clicking on the desired object and then selecting the **Delete** action.



Diagnosing a Database



The command ZOMDiagnose reports any improper or unusual object definitions in the Data Dictionary.

```
ZOMDiagnose [;d AZimDocument]
```

Associated with the option “;d”, it saves the report on that specific Zim document.



ZIM 9.10

Basic ZOM