Capgemini

ABAP Part II

Lesson 2: Object oriented ABAP

Lesson Objectives

After completing this lesson, participants will be able to understand -

- OOPS Concepts
- ABAP Objects
- Creating & Accessing objects
- Constructor
- Inheritance
- Casting
- Interfaces
- Events
- Exceptions

© 2018 Capgemini. All rights reserved.

OOPS Concept

Object-oriented programming, is a problem-solving method in which the software solution reflects objects in the real world.

Benefits of Object-oriented programming are :

- Multiple Instances
- Encapsulation
- Inheritance
- Polymorphism
- Compatibility
- Maintainability

© 2018 Capgemini. All rights reserved.

Benefits of Object-oriented programming

Multiple Instances

- The ability to create multiple instances of a "class", such as a vehicle, is one of the central attributes of object-oriented languages.

Encapsulation

- Encapsulation means that the implementation of an object is hidden from other components in the system, so that they cannot make assumptions about the internal status of the object and therefore dependencies on specific implementations do not arise

© 2018 Capgemini. All rights reserved.

Benefits of Object-oriented programming

Polymorphism

- Polymorphism (ability to have multiple forms) in the context of object technology signifies that objects in different classes react differently to the same messages.

Inheritance

- Inheritance defines the implementation relationship between classes, in which one class (the subclass) shares the structure and the behavior defined in one or more other classes (super classes).
- Note: ABAP Objects only allows single inheritance.

Compatibility

- ABAP object is true extension of ABAP language. ABAP OOPS statements can be used in procedural ABAP programs. Object themselves can contain classic ABAP statements, Only OOPS concepts that have been proved useful have been included. It has been kept the simplest. There is increased use of type checks.

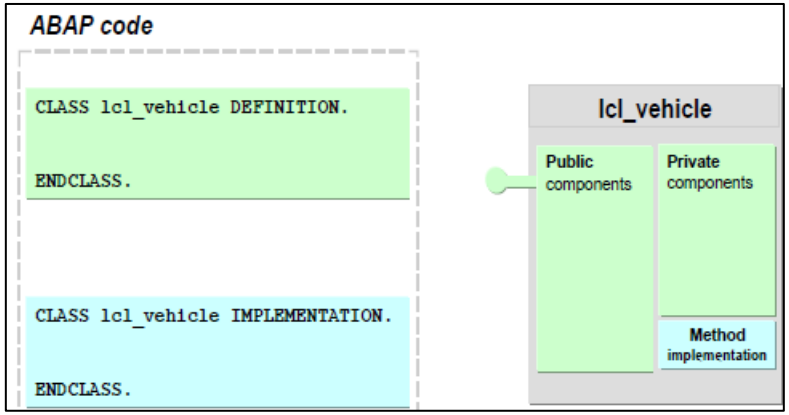
© 2018 Capgemini. All rights reserved.

ABAP Objects - Class

A class is a set of objects that have the same structure and the same behavior. A class is therefore like a blueprint, in accordance with which all objects in that class are created.

The components of the class are defined in the definition part. The components are attributes, methods, events, constants, types, and implemented interfaces. Only methods are implemented in the implementation part.

The CLASS statement cannot be nested, that is, you cannot define a class within a class.



© 2018 Capgemini. All rights reserved.

Classes

Classes are the central element of object-orientation.

A Class is an abstract description of an object.

Classes are templates for objects.

Defines the state and behavior of the object.

Types of classes

- Local classes
  - Defined within an ABAP program
  - Can be used only within that program
- Global classes
  - Defined in the class builder SE24
  - Stored centrally in class library in the R/3 repository
  - Can be accessed from all the programs in the R/3 system
  - e.g. CL_GUI_ALV_GRID, CL_GUI_CUSTOM_CONTAINER

© 2018 Capgemini. All rights reserved.

ABAP Objects - Attributes

Attributes describe the data that can be stored in the objects of a class.

Attributes can have any kind of data type:

- C, N, I, P, ..., STRING
- Dictionary types
- User-defined types
- TYPE REF TO defines a reference to an object, in this case “r_car”

Public attributes

- Can be viewed and changed by all users and in all methods
- Direct access

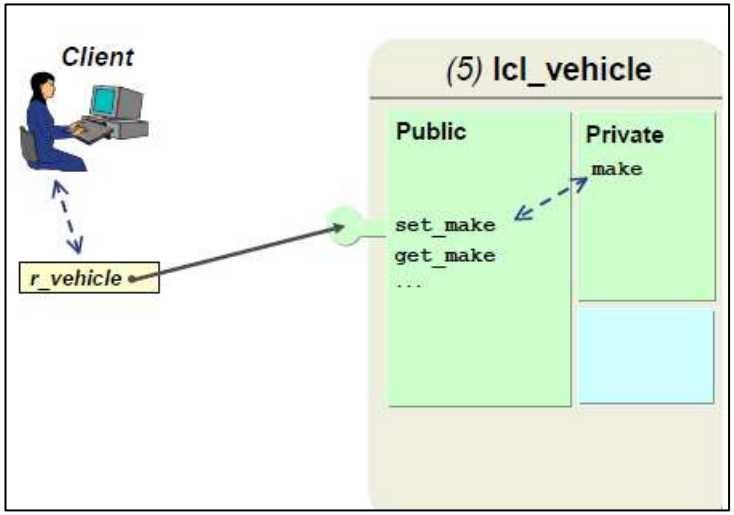
Private attributes

- Can only be viewed and changed from within the class
- No direct access from outside the class

© 2018 Capgemini. All rights reserved.

ABAP Objects - Attributes

Accessing private attributes :You can access an object's private attributes using public methods, which in turn output this attribute or change it.




```
CLASS lcl_vehicle DEFINITION.

  PUBLIC SECTION.
    DATA: make TYPE string.

  PRIVATE SECTION.

ENDCLASS.
```

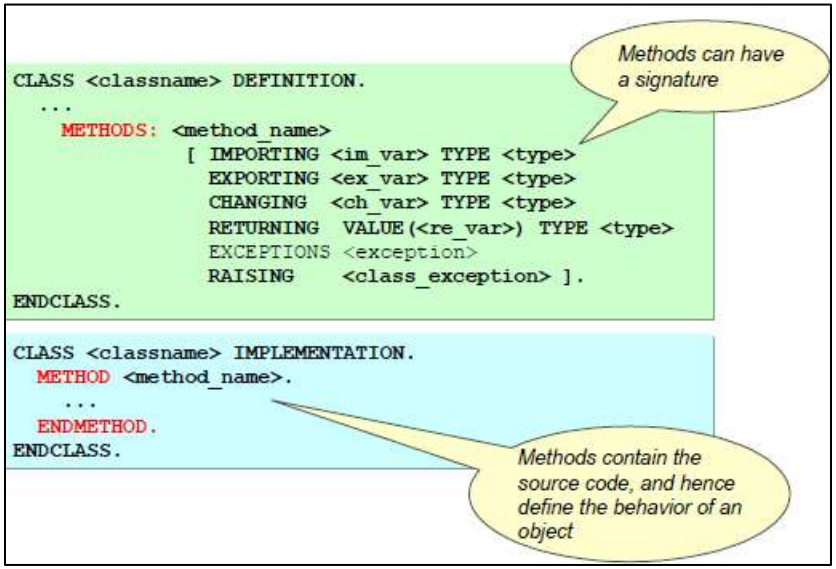
```
CLASS lcl_vehicle DEFINITION.

  PUBLIC SECTION.
    ...

  PRIVATE SECTION.
    DATA: make      TYPE string.

ENDCLASS.
```

© 2018 Capgemini. All rights reserved.

ABAP Objects - Methods

1. Methods are internal procedures in classes that determine the behavior of an object. They can access all attributes in their class and can therefore change the state of an object.

2. Methods have a parameter interface (called signature) that enables them to receive values when they are called and pass values back to the calling program.



© 2018 Capgemini. All rights reserved.

Methods and Visibility

Public methods

- Can be called from anywhere

Private methods

- Can only be called within the class

© 2018 Capgemini. All rights reserved.

Demo: Create a class with instance attributes and instance methods

© 2018 Capgemini. All rights reserved.

Instance attributes

Instance attributes

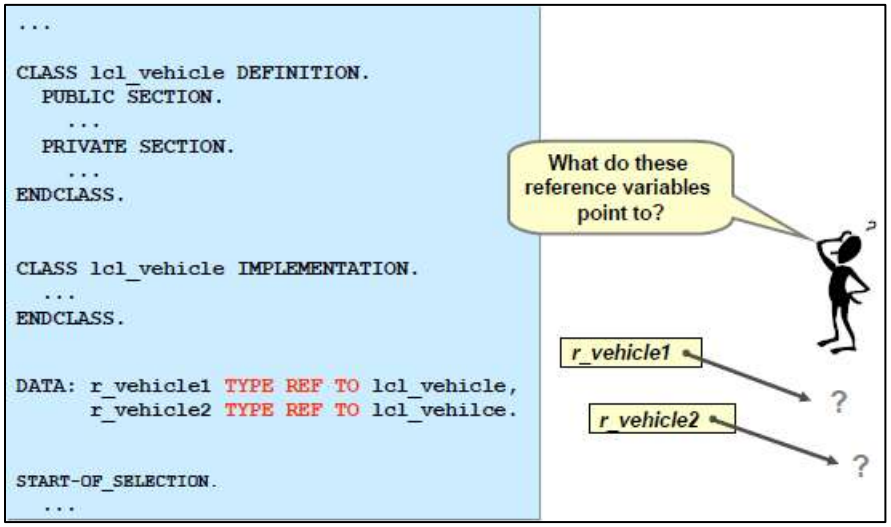
- One per instance
- Statement: DATA

© 2018 Capgemini. All rights reserved.

Reference Variable

A reference variable acts as a pointer to an object.

- DATA: R_VEHICLE1 TYPE REF TO LCL_VEHICLE.
- Declares a reference variable that acts as a pointer to an object



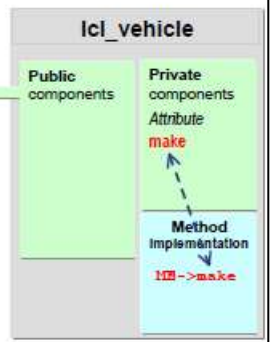
© 2018 Capgemini. All rights reserved.

Reference Variable ME

You can address the object itself within instance methods using the implicitly available reference variable me.

Description of example: In the constructor, the instance attribute make is covered by the locally defined variable make. In order to still be able to address the instance attribute, you need to use me.

```
CLASS lcl_vehicle DEFINITION.
  PUBLIC SECTION.
    METHODS constructor
       IMPORTING im_make  TYPE string
                 im_modell TYPE string.
  PRIVATE SECTION.
    DATA make TYPE string. ...
ENDCLASS.

CLASS lcl_vehicle IMPLEMENTATION.
  METHOD constructor.  "just a demo
    DATA make TYPE string VALUE 'Benz'.
    CONCATENATE im_make make INTO ME->make.
  ENDMETHOD.

  METHOD DUMMY.
* calling own methods from inside class
* call method display_attributes( )
* with the short syntax variant:
    display_attributes( ).
  ENDMETHOD.
ENDCLASS.
```



© 2018 Capgemini. All rights reserved.

Demo: Create a class with Reference variable Me

© 2018 Capgemini. All rights reserved.

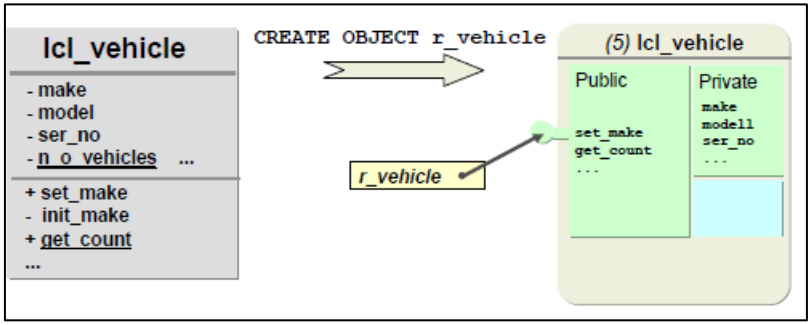
Creating and Accessing Objects

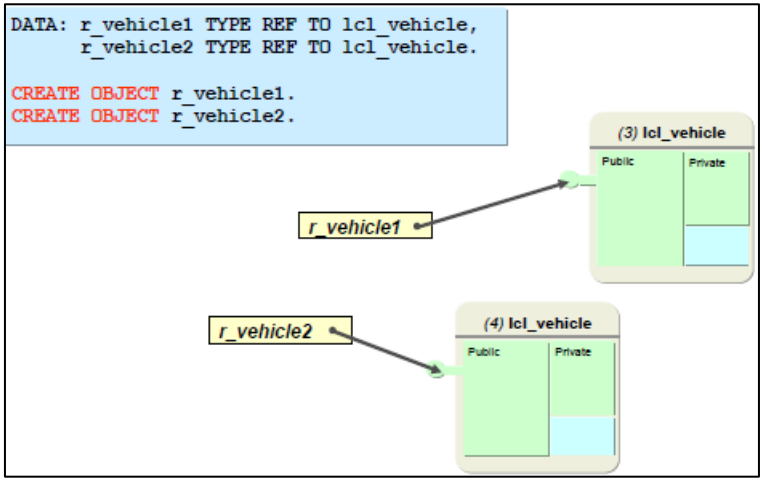
An object is a section of source code that contains data and provides services

Objects are created using the CREATE OBJECT statement

Objects can only be created and addressed using reference variables

An object is a section of source code that contains data and provides services.





© 2018 Capgemini. All rights reserved.

Accessing Attributes and Method

Instance methods are called using

CALL METHOD <reference>-><instance_method>.

Static methods (also referred to as class methods) are called using

CALL METHOD <classname>=><class_method>.

If you are calling a static method from within the class, you can omit the class name.

Static attributes are accessed using

<classname>=><class_attribute>

instance attributes are accessed using

<instance>-><instance_attribute>

=> and -> are the component selectors

© 2018 Capgemini. All rights reserved.

Constructor

The constructor is a special instance method in a class with the name constructor.

Each class can have one constructor.

The constructor is automatically called at runtime within the CREATE OBJECT statement.

If you need to implement the constructor, then you must define and implement it in the PUBLIC SECTION.

You cannot normally call the constructor explicitly

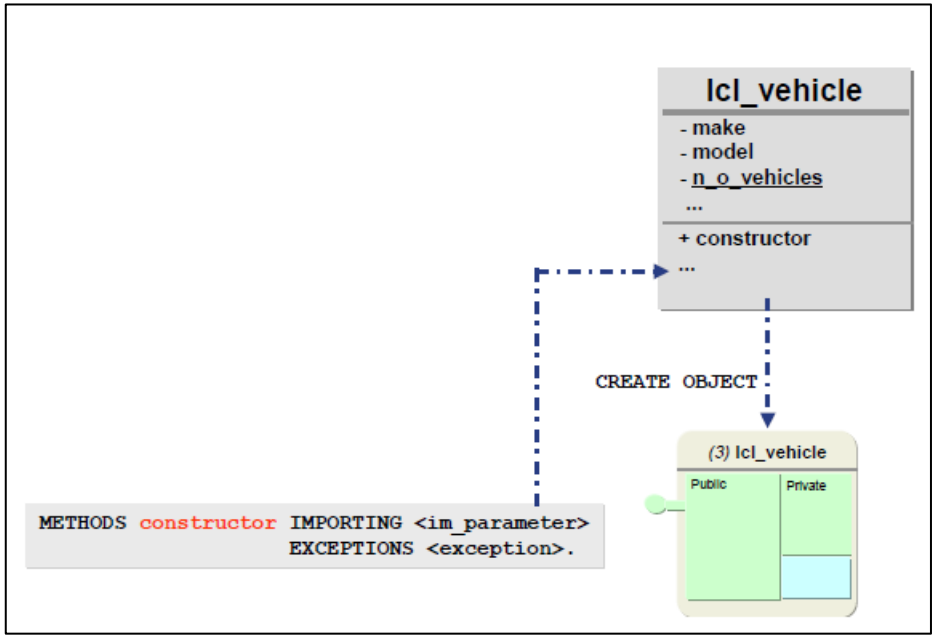
Special method for creating objects with defined initial state

Only has IMPORTING parameters and EXCEPTIONS

Is executed only once per instance

© 2018 Capgemini. All rights reserved.

Constructor



© 2018 Capgemini. All rights reserved.

Demo: Create a class with Constructor

© 2018 Capgemini. All rights reserved.

Demo: Global Class creation through SE24

© 2018 Capgemini. All rights reserved.

Summary

In this lesson, you have learnt:

- OOPS Concepts
- ABAP Objects
- Creating & Accessing objects
- Constructor

© 2018 Capgemini. All rights reserved.

Review Question

Question 1: _____ do not contain implementation.

Question 2: _____means that the implementation of an object is hidden from other components in the system.

© 2018 Capgemini. All rights reserved.