

In this video, I want to discuss constructor's again, but this time a different type of constructor, and this is called a static constructor.

So a static constructor is very similar to a normal constructor, but it is executed the first time either an object of a class is created, either a static attribute of a class is used or a static method of a class is used.

Remember, when we create class data, well, we can also create class methods. And in our case here, we're looking at creating a static constructor.

So there are a few rules as always.

And these are we must declare, the static constructor in the public section of our class.

There is never an interface for our static constructor, so we don't use any parameters at all.

And then a static constructor can only access static attributes of their class.

And if you think about it, this sort of makes sense, because if we have instantiated an object, then attributes of an object will not be filled in.

They will not exist with static data that static data can be accessed without an object being instantiated.

So for a static constructor, it sort of makes sense that the only thing it can really touch that exists in the class are those static attributes, that class data.

So when we refer to the static constructor, when we want to code it, it's important that we remember that this type constructor is called a class constructor, as you can see here, plus constructor.

And the syntax would be a class method.

If you remember back to when we declare class attributes, we say class-data what this one is class-methods and then we call it the class underscore constructor.

It's a fixed name, we must call it that.

METHOD CLASS\_CONSTRUCTOR.  
    CARLOG = 'Car class has been used class constructor'.  
    WRITE: / CARLOG.  
  ENDMETHOD.

Static constructor.

Then we need to move down to the implementation section and put in the code.

So I use the methods keyword of a method, then a class constructor and as an example, all I'm going to do is just write out some checks to our screen to indicate that this class constructor has been executed.

Class has been used, so just set a date right there and I'm going to write out the catalogue to the screen.

And now this class also at this constructor is going to be executed, if you remember, back to the

slide in our code when we create an object.

But if we just have any code that just referenced the class data, it will get executed then as well.

REPORT ZYNY\_CLASS\_00.  
  
CLASS CAR DEFINITION.  
  PUBLIC SECTION.  
    CLASS-DATA: NUMOFCARS TYPE I.           *"Static Arrtibute*  
  
    CLASS-METHODS CLASS\_CONSTRUCTOR.        *"Static-Constructor*  
  
    METHODS CONSTRUCTOR                     *"Instance Constructor*  
      IMPORTING  
        MAKE TYPE C  
        MODEL TYPE C  
        NUMSEATS TYPE I  
        MAXSPEED TYPE I.  
  
    METHODS VIEWCAR.  
  
    METHODS SETNUMSEATS  
      IMPORTING  
        NEWSEATNUM TYPE I.  
*\*     EXPORTING                             "Placed all 4 on screen to show the options we have. We selected IMPORTING*  
*\*     CHANGING*  
*\*     RETURNING*  
  
    METHODS GOFASTER  
      IMPORTING  
        INCREMENT TYPE I  
      EXPORTING  
        RESULT TYPE I.  
  
    METHODS GOSLOWER                        *"Just as an example we have defined this method using RETURNING.*  
      IMPORTING                             *"It can be exactly the same as the GOFASTER method - Importing & Exporting*  
        INCREMENT TYPE I  
      RETURNING  
        VALUE(RESULT) TYPE I.  
  
  
  
  PRIVATE SECTION.  
    DATA: MAKE TYPE C LENGTH 20,            *"Have to use a METHOD to read or change it. It could go into the PUBLIC section*  
                                            *"and be defined as: data: make type c LENGTH 20 READ-ONLY. This would allow other*  
                                            *"objects to read it but not change it due to the READ-ONLY addition.*  
          MODEL TYPE C LENGTH 20,  
          NUMSEATS TYPE I,  
          SPEED TYPE I,  
          MAXSPEED TYPE I.  
  
  CLASS-DATA: carlog TYPE c LENGTH 40.    *"Used by the Class\_Constructor*  
  
ENDCLASS.                    *"car DEFINITION*  
  
*\*----------------------------------------------------------------------\**  
*\*       CLASS car IMPLEMENTATION*  
*\*----------------------------------------------------------------------\**  
*\**  
*\*----------------------------------------------------------------------\**  
CLASS CAR IMPLEMENTATION.  
  
  METHOD CLASS\_CONSTRUCTOR.  
    CARLOG = 'Car class has been used class constructor'.  
    WRITE: / CARLOG.  
  ENDMETHOD.                    *"class\_constructor*  
  
   METHOD CONSTRUCTOR.                       *"Instance Constructor*  
     write : / 'normal constructor'.  
*\*    make = make.                           "Run a syntax check and notice the error. We have an ATTRIBUTE & PARAMETER with the same name*  
*\*    model = model.                         "This is where we can use the SELF-Reference notation  me->attribute\_name*  
*\*    numseats = numseats.*  
*\*    maxspeed = maxspeed.*  
    ME->MAKE = MAKE.                        *"Add the ME-> after the slides*  
    ME->MODEL = MODEL.  
    ME->NUMSEATS = NUMSEATS.  
    ME->MAXSPEED = MAXSPEED.  
    NUMOFCARS = NUMOFCARS + 1.              *"Increment the car counter Static Variable*  
  ENDMETHOD.                    *"constructor*  
  
  METHOD VIEWCAR.                           *" Add this as a check and DEBUG the code.*  
    WRITE: / 'Make = ', 19 MAKE.  
    WRITE: / 'Model = ', 19 MODEL .  
    WRITE: / 'Number of Seats = ', 19 NUMSEATS LEFT-JUSTIFIED.  
    WRITE: / 'Max Speed = ', 19 MAXSPEED LEFT-JUSTIFIED.  
    WRITE: / 'Speed = ', 19 SPEED LEFT-JUSTIFIED.  
  ENDMETHOD.                    *"viewcar*  
  
  METHOD SETNUMSEATS.  
    NUMSEATS = NEWSEATNUM.  
  ENDMETHOD.                    *"setnumseats*  
  
  METHOD GOFASTER.  
    DATA  TMPSPEED TYPE I.  
    TMPSPEED = SPEED + INCREMENT.  
    IF TMPSPEED <= MAXSPEED.  
      SPEED = SPEED + INCREMENT.  
    ENDIF.  
    RESULT = SPEED.  
  ENDMETHOD.                    *"gofaster*  
  
  METHOD GOSLOWER.  
    DATA  TMPSPEED TYPE I.  
    TMPSPEED = SPEED - INCREMENT.  
    IF TMPSPEED >= 0.  
      SPEED = SPEED - INCREMENT.  
    ENDIF.  
    RESULT = SPEED.  
  ENDMETHOD.                    *"goslower*  
  
ENDCLASS.                    *"car IMPLEMENTATION*  
  
*\* Our program starts here.*  
  
START-OF-SELECTION.             *"We need to tell the system where to start our program..*  
  
  DATA CAR1 TYPE REF TO CAR.    *"Define an object reference variable. Notice how we have to use TYPE REF TO*  
  CREATE OBJECT CAR1  
    EXPORTING  
      MAKE     = 'AUDI'  
      MODEL    = 'A4'  
      NUMSEATS = 5  
      MAXSPEED = 120.  
  
  DATA CAR2 TYPE REF TO CAR.    *"Define an object reference variable. Notice how we have to use TYPE REF TO*  
  CREATE OBJECT CAR2  
    EXPORTING  
      MAKE     = 'AUDI'  
      MODEL    = 'A4'  
      NUMSEATS = 5  
      MAXSPEED = 120.  
  
*\*  CAR1=>VIEWCAR( ).*