VARIAN SCRIPTING EXERCISE 1

Developer Workshop 2.0 – Austin, Texas – July 18th, 2014

VARIAN
ONCOLOGY
SYSTEMS

Wayne Keranen

Product Manager, Varian APIs

July 18th, 2014

VARIAN
medical systems

Disclaimers

- Eclipse^{TM} and ARIA^{TM} are trademarked by Varian Medical Systems.
- Word^{TM}, Excel^{TM}, Office^{TM} are trademarked by Microsoft.
- Visual Studio^{TM} is trademarked by Microsoft.

VARIAN medical systems

Exercise 1 Overview

Hands-on exercise guides learners through development of a simple single file plugin script that calculates and exports the DVH for the loaded plan and the PTV.

Case “exercise1”, plan ‘4FldBox’.

Sign in to Virtual Eclipse Environment

- Before we start, sign in with your assigned userid/pwd to your assigned Eclipse Client.
- TBD

Two kinds of Eclipse scripts

- Eclipse calls you - **Plugin**

- You call Eclipse - **Standalone Executable**

  (Standalone Executable - “An Application”.

  Examples: Microsoft Word, Excel)

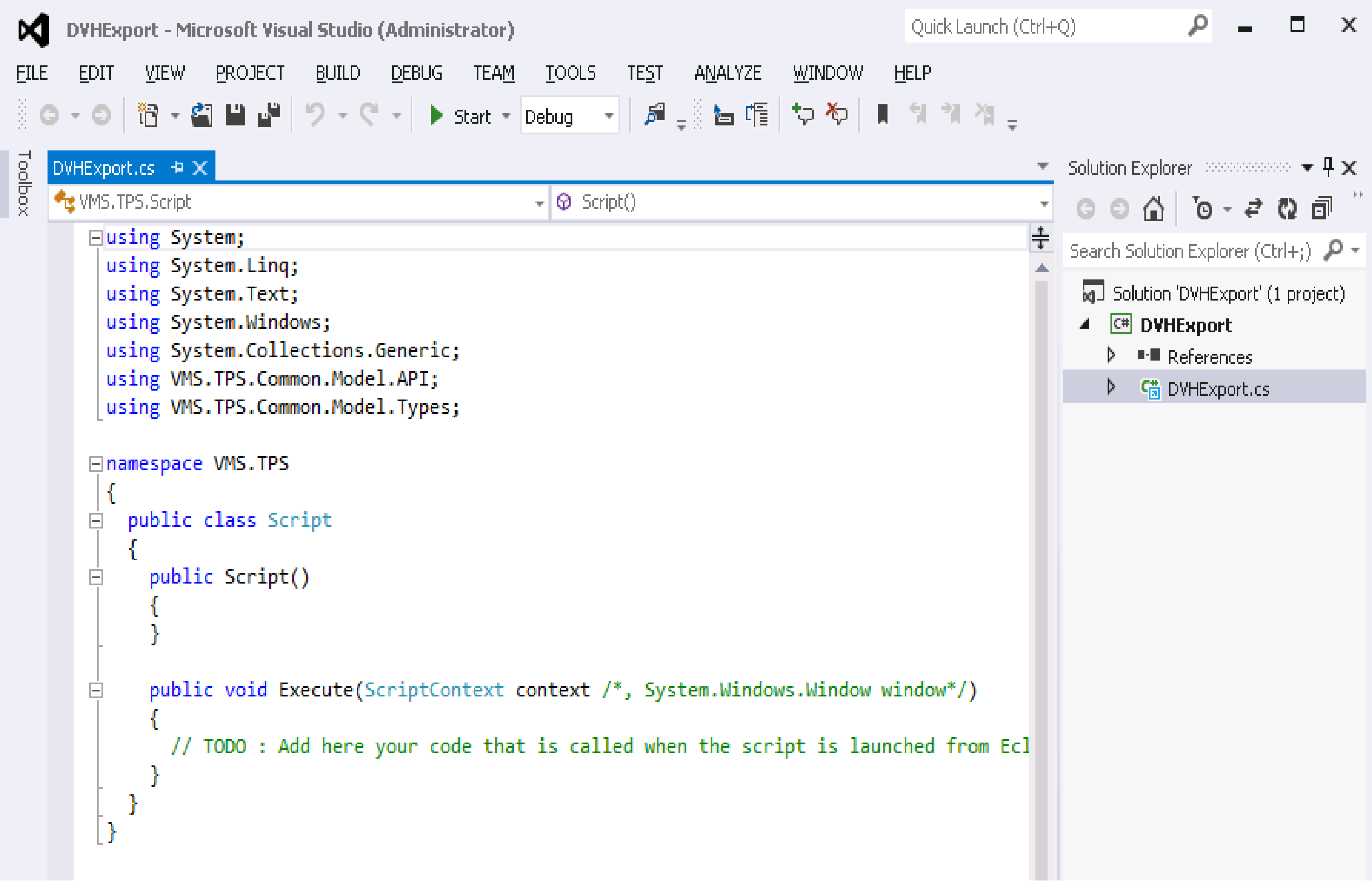
Exercise 1 – DVH Export Plugin Script

- Step 1: Get into Eclipse External Beam.

  ARIA userid/pwd: **allrights/allrights**

- Step 2: Load the patient ‘exercise1’

- Step 3: Run Script Wizard and open Online Help. Create a Plugin Script and name it “DVHExport”, Open project in Visual Studio.


DVHExport - Microsoft Visual Studio (Administrator)
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Start
Debug
Toolbox
DVHExport.cs
VMS.TPS.Script
Script()
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'DVHExport' (1 project)
DVHExport
References
DVHExport.cs

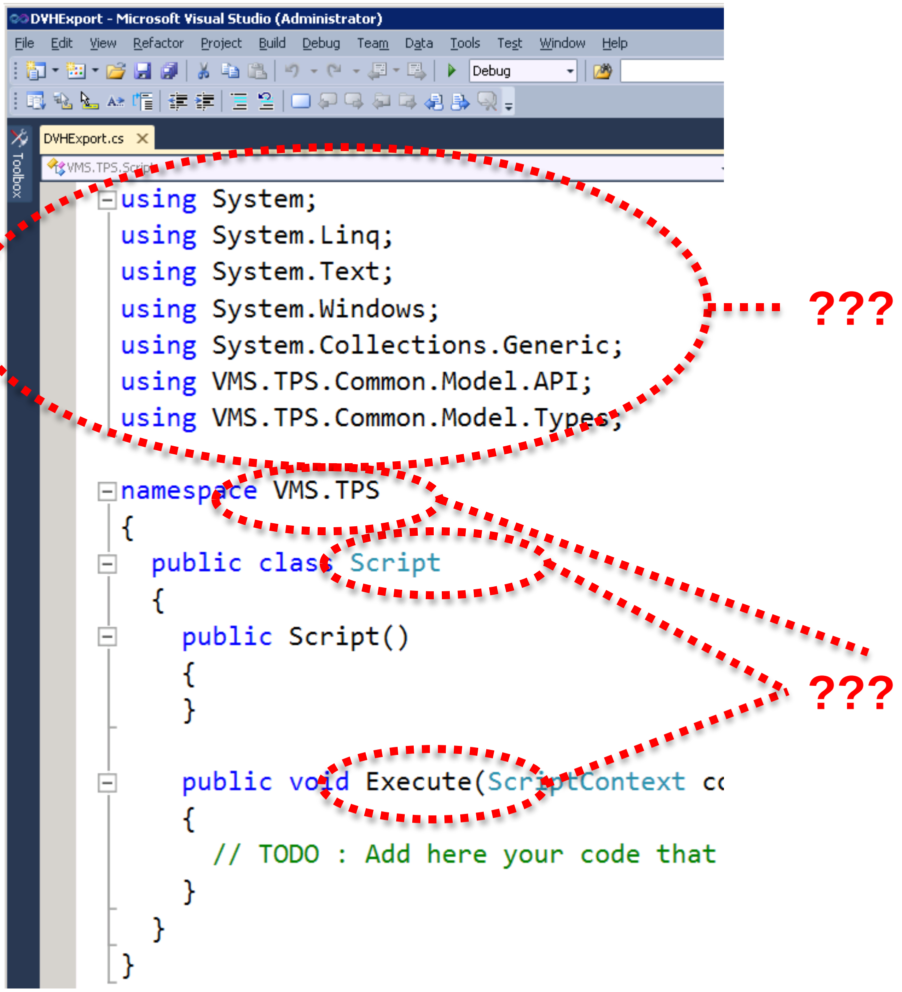
```
using System;
using System.Linq;
using System.Text;
using System.Windows;
using System.Collections.Generic;
using VMS.TPS.Common.Model.API;
using VMS.TPS.Common.Model.Types;

namespace VMS.TPS
{
  public class Script
  {
    public Script()
    {
    }

    public void Execute(ScriptContext context /*, System.Windows.Window window*/)
    {
      // TODO : Add here your code that is called when the script is launched from Ecl
    }
  }
}
```

VARIAN medical systems

Plugin script - C# Syntax Notes



DVHExport - Microsoft Visual Studio (Administrator)
File Edit View Refactor Project Build Debug Team Data Tools Test Window Help
Debug
Toolbox
DVHExport.cs
VMS.TPS.Scri
???
???

```
using System;
using System.Linq;
using System.Text;
using System.Windows;
using System.Collections.Generic;
using VMS.TPS.Common.Model.API;
using VMS.TPS.Common.Model.Types;

namespace VMS.TPS
{
  public class Script
  {
    public Script()
    {
    }

    public void Execute(ScriptContext cc
    {
      // TODO : Add here your code that
    }
  }
}
```

C# imports - similar to C++ ‘#include’, java & python ‘import’.

Ignore for now.

<u>When loading a Plugin Script:</u>

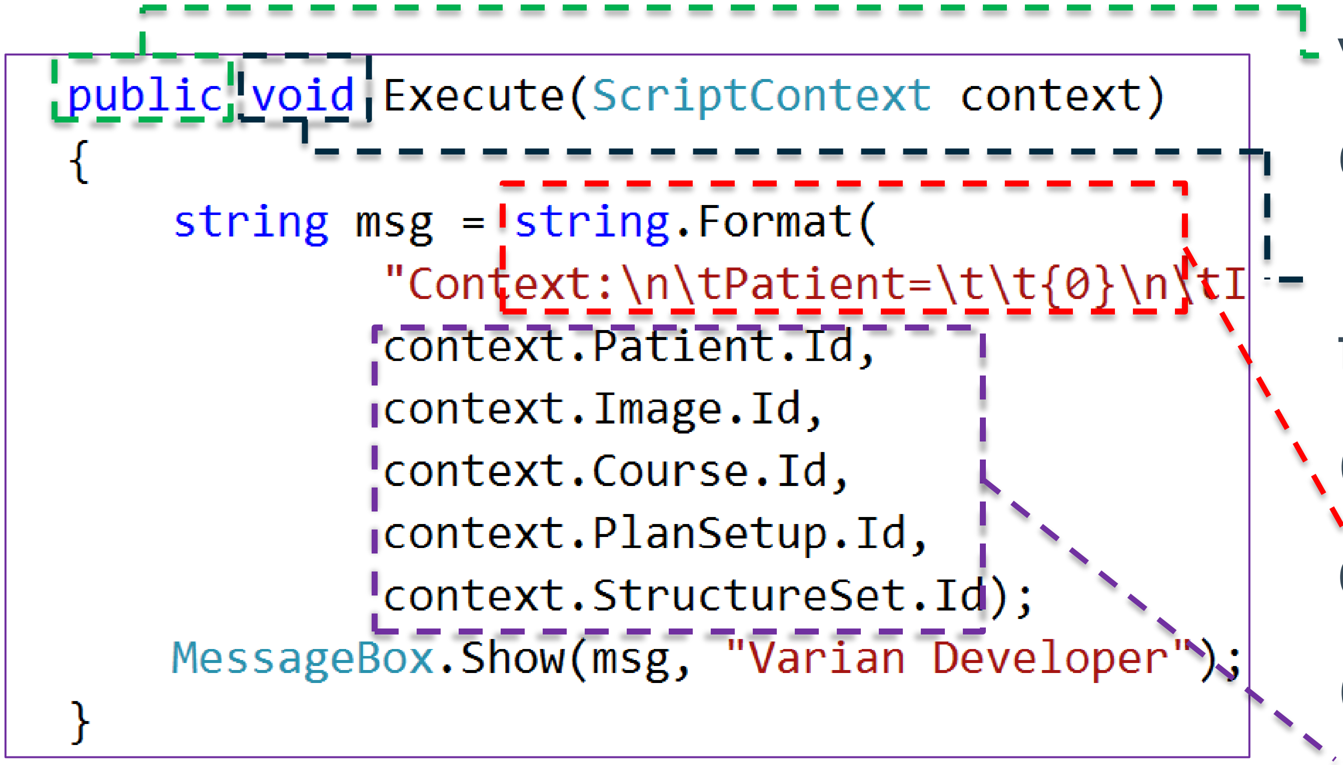
Eclipse looks for class “Script” in namespace “VMS.TPS” (VMS.TPS.Script), and tries to call method “Execute” and pass it a ScriptContext that Eclipse has created and populated.

Exercise 1 continued…

- Step 4: Select below `//TODO` in code file, then insert code snippet dw -> exercise1 -> Step 4.

- Step 5. F6 to Compile then run script in Eclipse.

Some C# Syntax



```
public void Execute(ScriptContext context)
{
    string msg = string.Format(
            "Context:\n\tPatient=\t\t{0}\n\tI
            context.Patient.Id,
            context.Image.Id,
            context.Course.Id,
            context.PlanSetup.Id,
            context.StructureSet.Id);
    MessageBox.Show(msg, "Varian Developer");
}
```

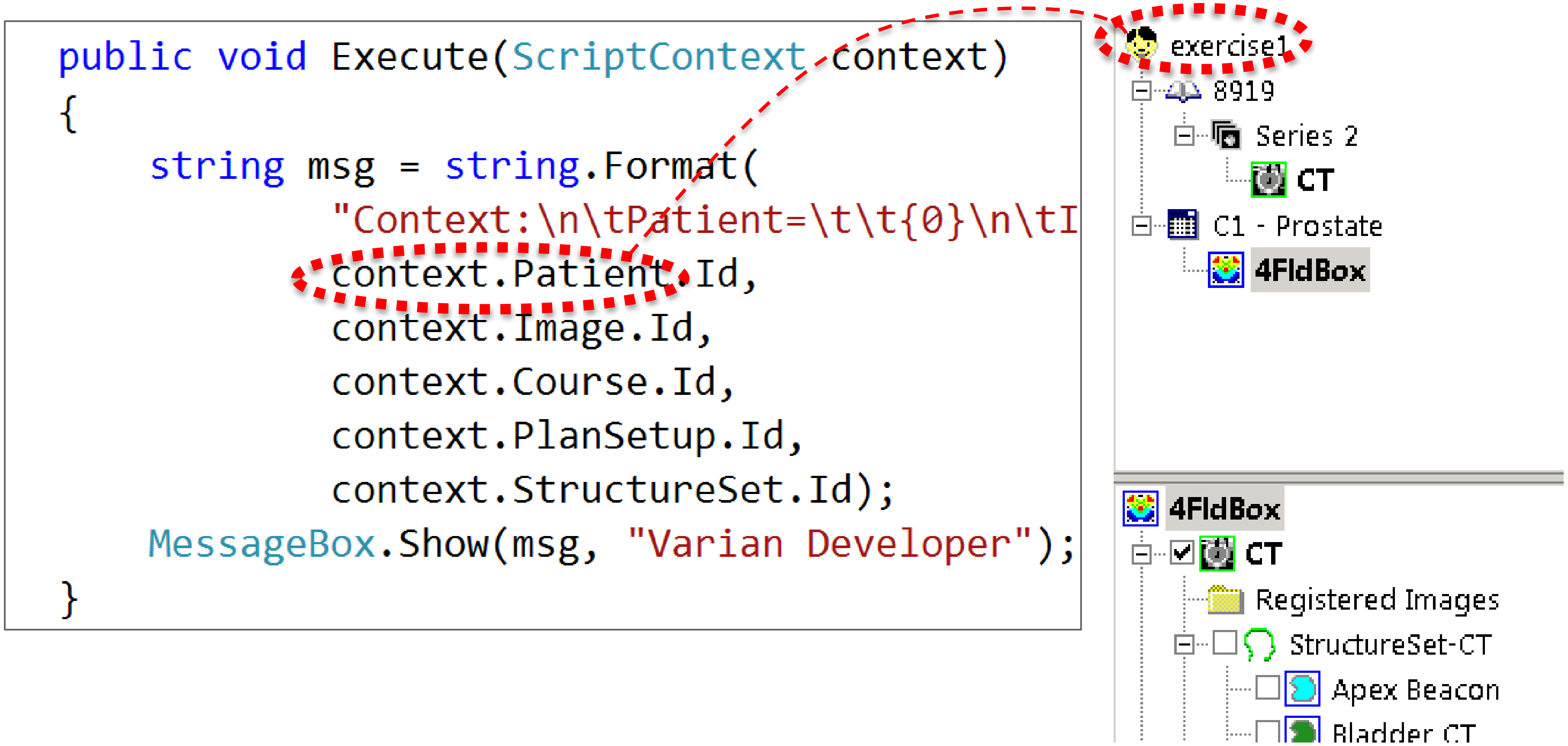
Visibility of class method

Method return type.

C# version of 'sprintf'.

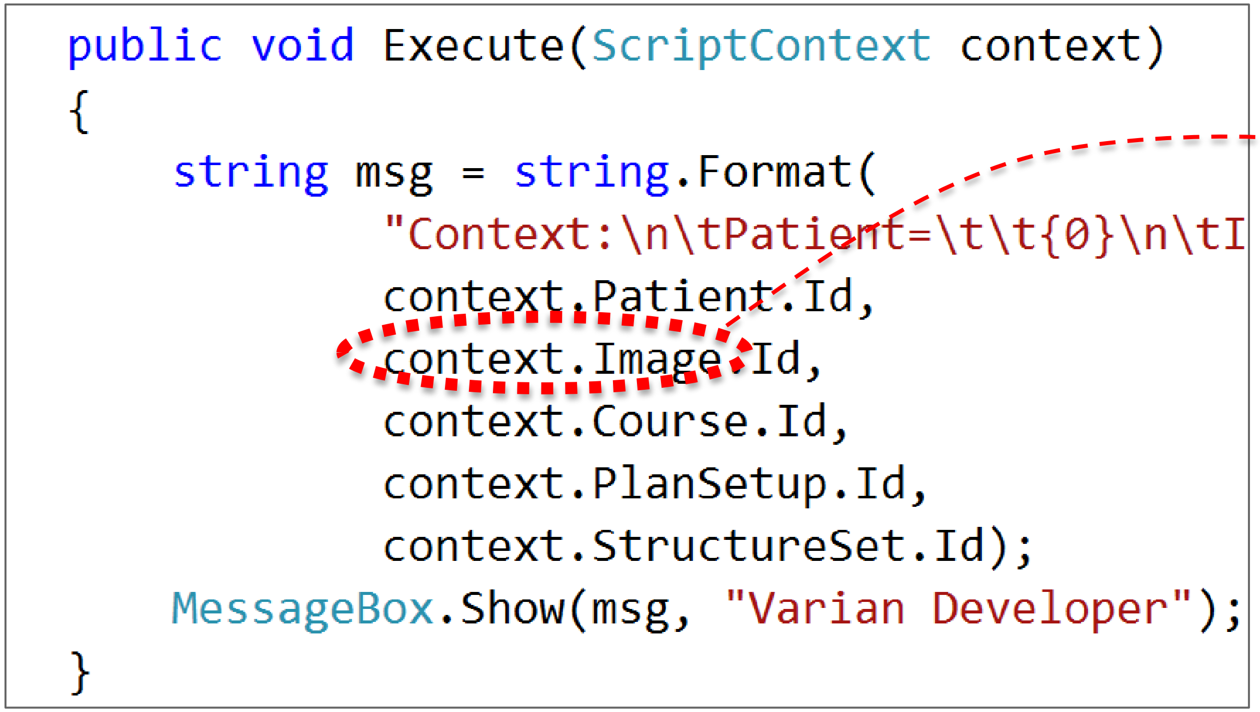
C# properties.

Plugin Script Context

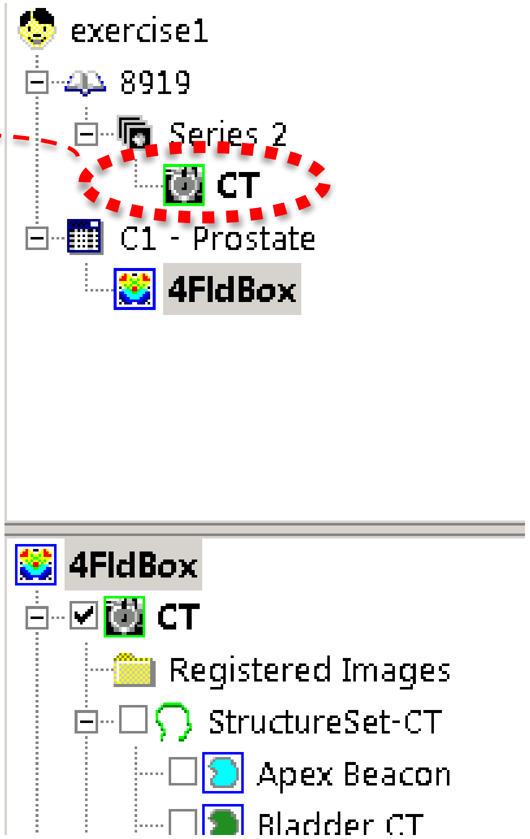


```
public void Execute(ScriptContext context)
{
    string msg = string.Format(
            "Context:\n\tPatient=\t\t{0}\n\tI
            context.Patient.Id,
            context.Image.Id,
            context.Course.Id,
            context.PlanSetup.Id,
            context.StructureSet.Id);
    MessageBox.Show(msg, "Varian Developer");
}
```

Plugin Script Context



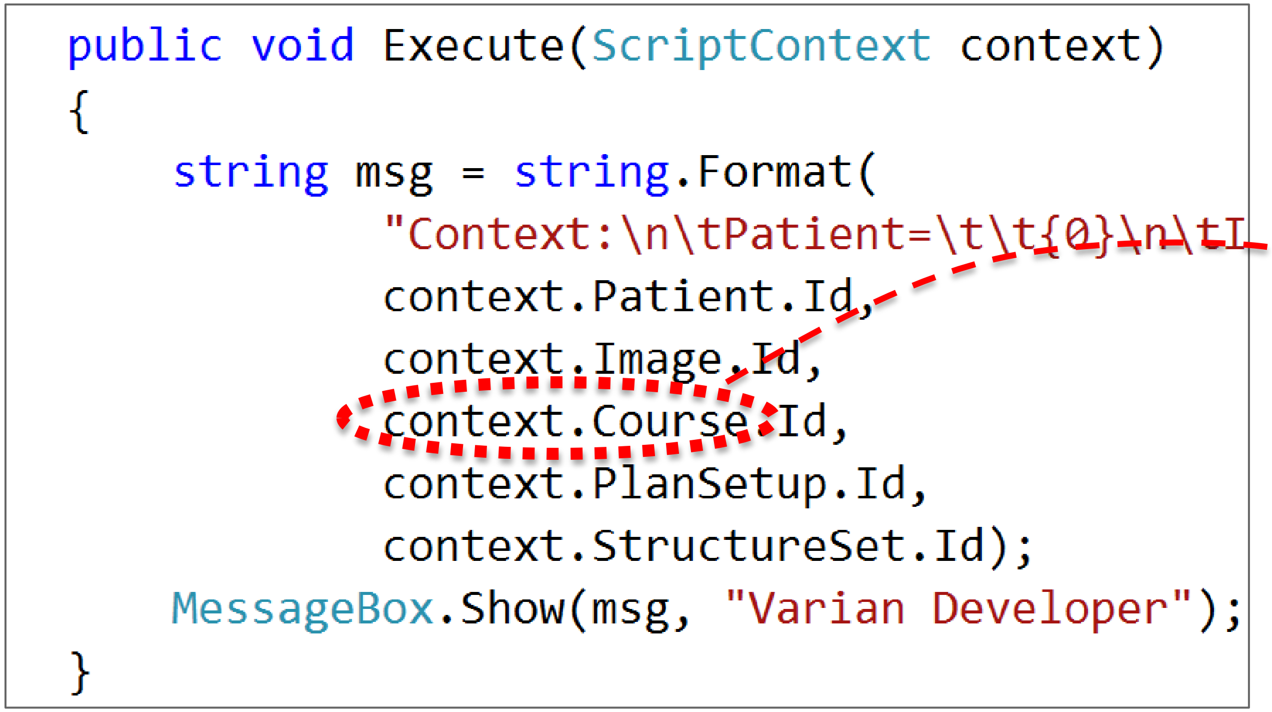
```
public void Execute(ScriptContext context)
{
    string msg = string.Format(
            "Context:\n\tPatient=\t\t{0}\n\tI
            context.Patient.Id,
            context.Image.Id,
            context.Course.Id,
            context.PlanSetup.Id,
            context.StructureSet.Id);
    MessageBox.Show(msg, "Varian Developer");
}
```



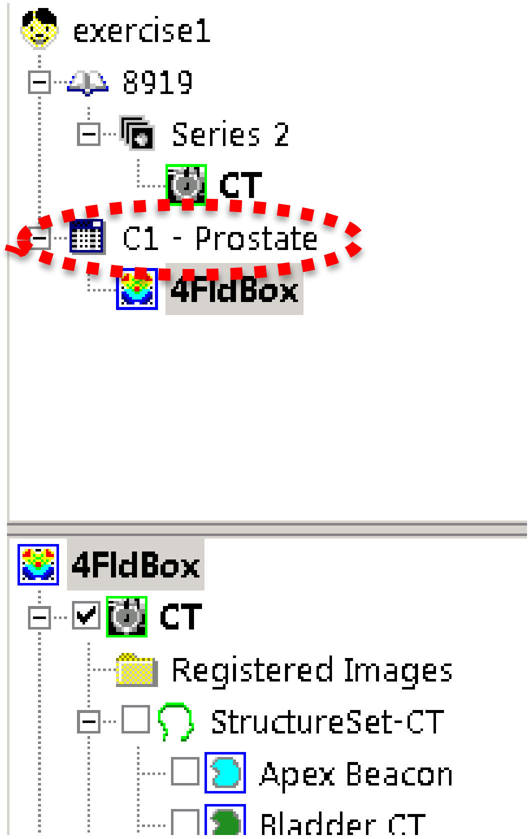
```
exercise1
⊟ 8919
  ⊟ Series 2
      CT
⊟ C1 - Prostate
    4FldBox

4FldBox
⊟ ☑ CT
    Registered Images
  ⊟ ☐ StructureSet-CT
      ☐ Apex Beacon
      ☐ Bladder CT
```

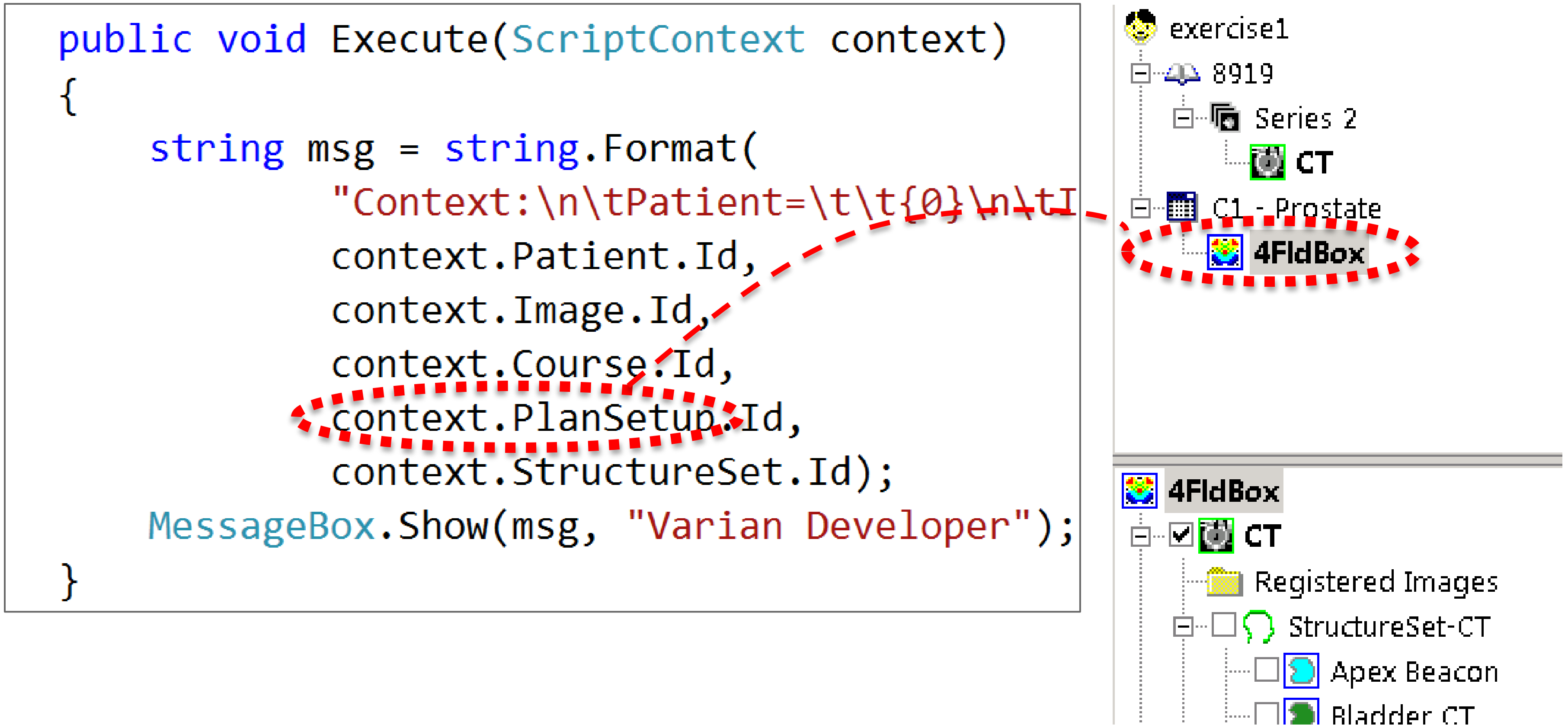
Plugin Script Context



```
public void Execute(ScriptContext context)
{
    string msg = string.Format(
            "Context:\n\tPatient=\t\t{0}\n\tI
            context.Patient.Id,
            context.Image.Id,
            context.Course.Id,
            context.PlanSetup.Id,
            context.StructureSet.Id);
    MessageBox.Show(msg, "Varian Developer");
}
```



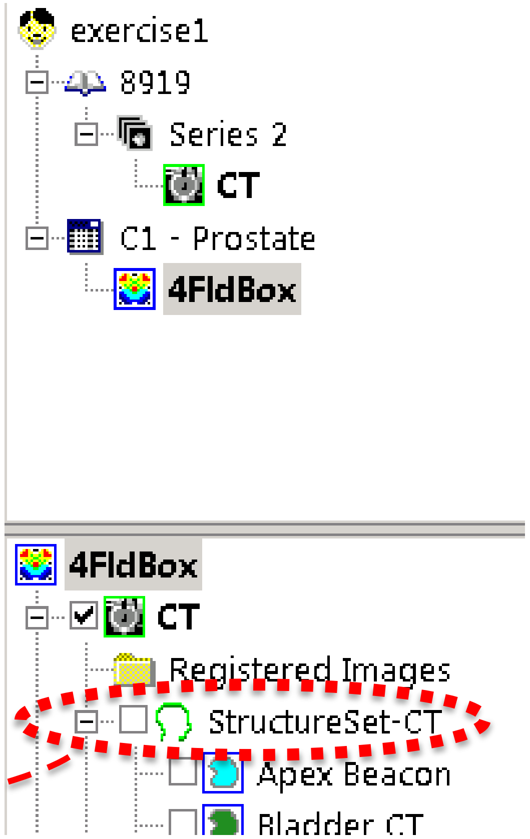
Plugin Script Context



```
public void Execute(ScriptContext context)
{
    string msg = string.Format(
            "Context:\n\tPatient=\t\t{0}\n\tI
            context.Patient.Id,
            context.Image.Id,
            context.Course.Id,
            context.PlanSetup.Id,
            context.StructureSet.Id);
    MessageBox.Show(msg, "Varian Developer");
}
```

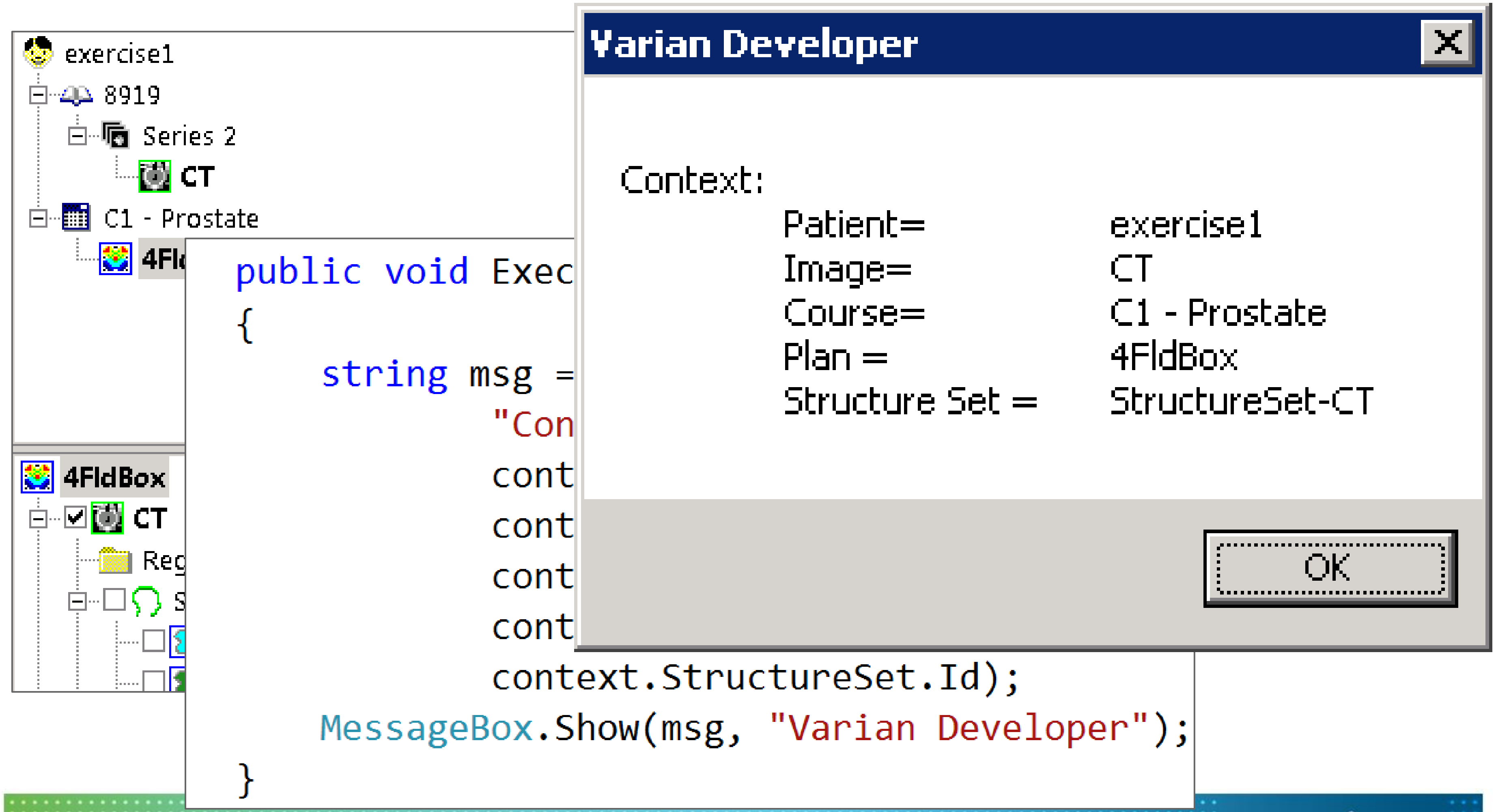
Plugin Script Context

```
public void Execute(ScriptContext context)
{
    string msg = string.Format(
            "Context:\n\tPatient=\t\t{0}\n\tI
            context.Patient.Id,
            context.Image.Id,
            context.Course.Id,
            context.PlanSetup.Id,
            context.StructureSet.Id);
    MessageBox.Show(msg, "Varian Developer");
}
```



VARIAN medical systems

Eclipse calls you : Plugin Script Context



```
public void Exec
{
        string msg =
                    "Con
                    cont
                    cont
                    cont
                    cont
                    context.StructureSet.Id);
        MessageBox.Show(msg, "Varian Developer");
}
```

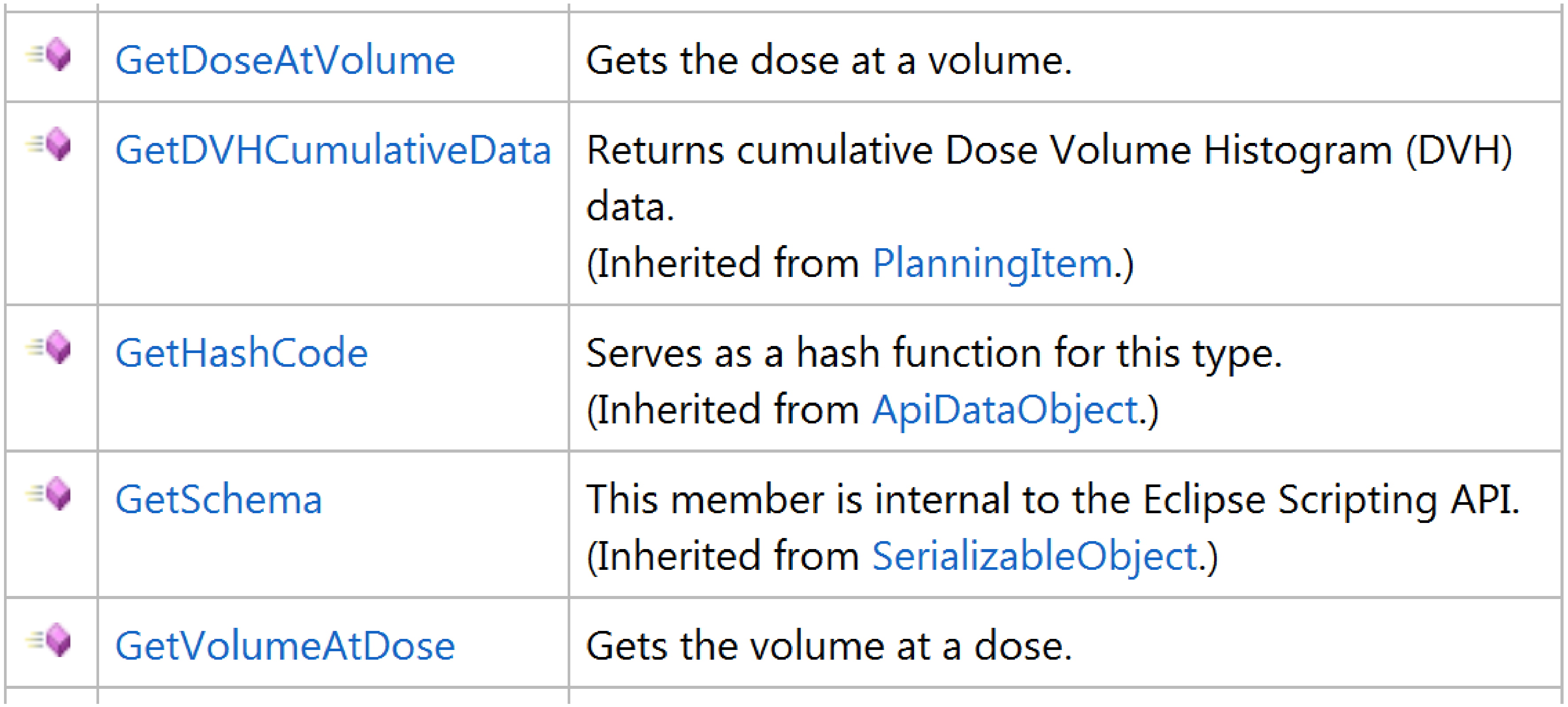
Varian Developer

Context:

Patient=	exercise1
Image=	CT
Course=	C1 - Prostate
Plan =	4FldBox
Structure Set =	StructureSet-CT

Extracting DVHs

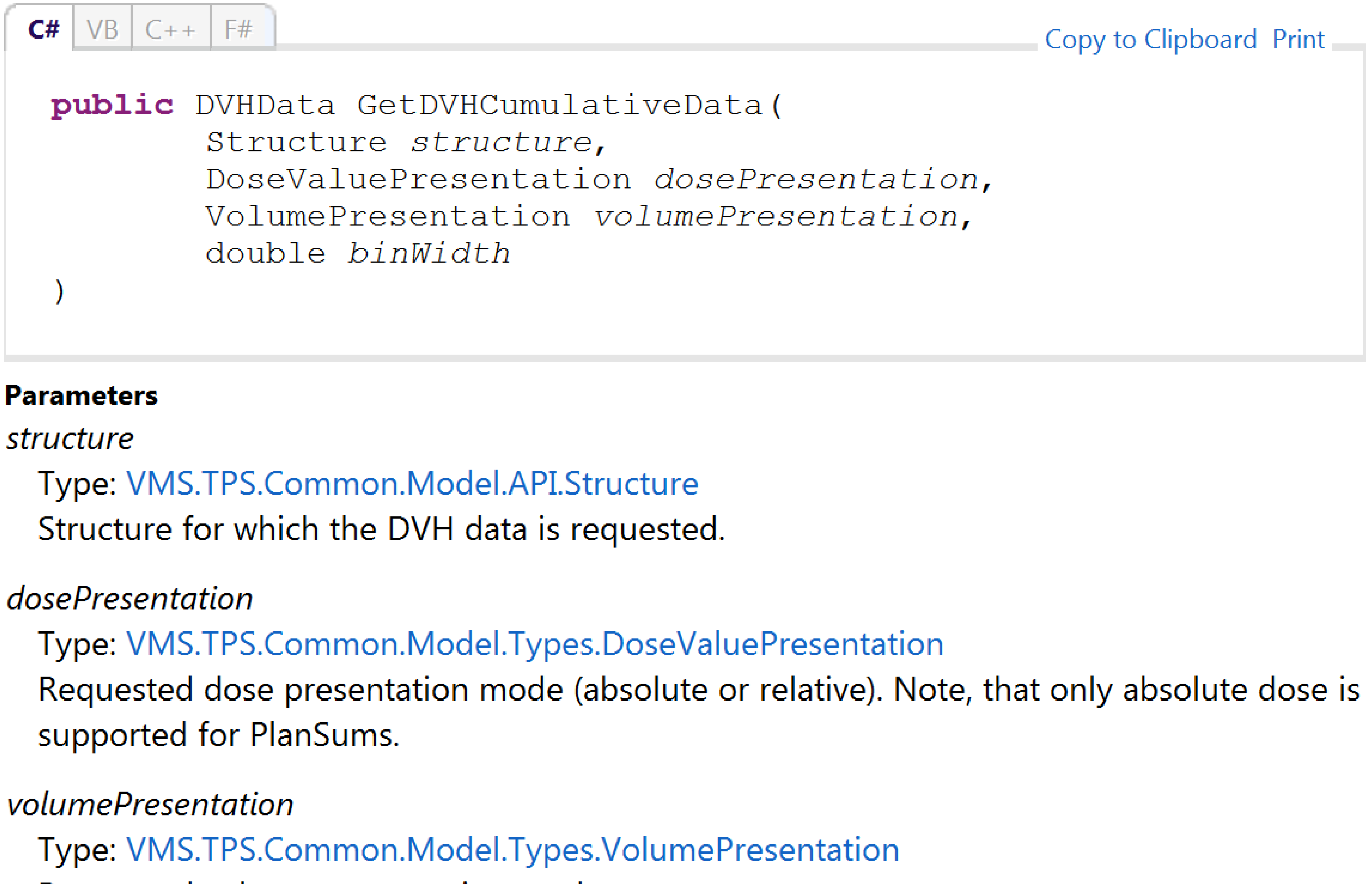
Consult ESAPI Online Help, PlanSetup class.



GetDoseAtVolume	Gets the dose at a volume.
GetDVHCumulativeData	Returns cumulative Dose Volume Histogram (DVH) data. (Inherited from PlanningItem.)
GetHashCode	Serves as a hash function for this type. (Inherited from ApiDataObject.)
GetSchema	This member is internal to the Eclipse Scripting API. (Inherited from SerializableObject.)
GetVolumeAtDose	Gets the volume at a dose.

VARIAN medical systems

Extracting DVHs…



```
public DVHData GetDVHCumulativeData(
        Structure structure,
        DoseValuePresentation dosePresentation,
        VolumePresentation volumePresentation,
        double binWidth
)
```

Parameters

structure

Type: VMS.TPS.Common.Model.API.Structure

Structure for which the DVH data is requested.

dosePresentation

Type: VMS.TPS.Common.Model.Types.DoseValuePresentation

Requested dose presentation mode (absolute or relative). Note, that only absolute dose is supported for PlanSums.

volumePresentation

Type: VMS.TPS.Common.Model.Types.VolumePresentation

Extracting DVHs

- So… we need a PlanSetup, and a Structure.

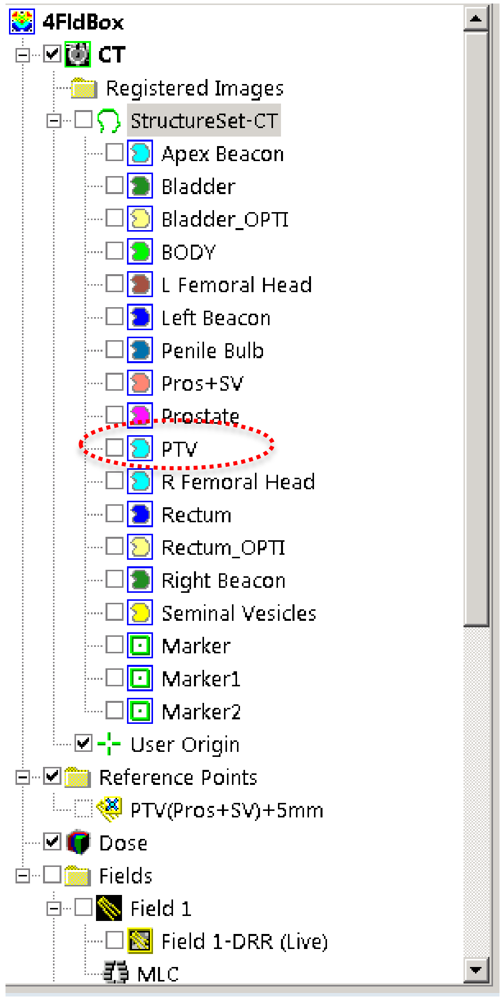
- PlanSetup: directly from the ScriptContext.

```
    context.PlanSetup
```

- Structure?

VARIAN medical systems

Script Context – Finding a Structure



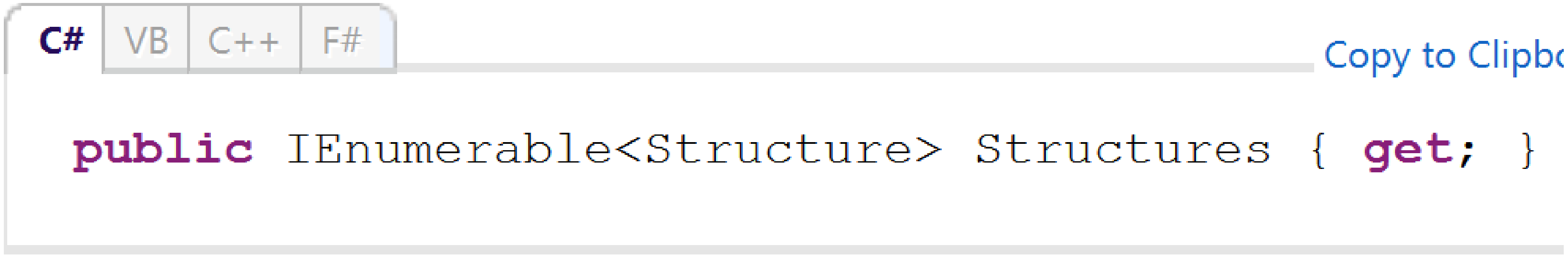
- Need to get a reference to the PTV structure.

- See Online Help for:

  ScriptContext
  .StructureSet.Structures

VARIAN medical systems

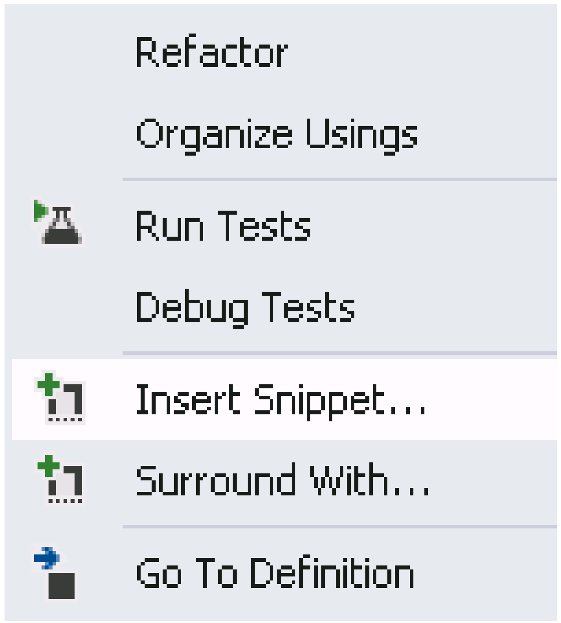
StructureSet.Structures

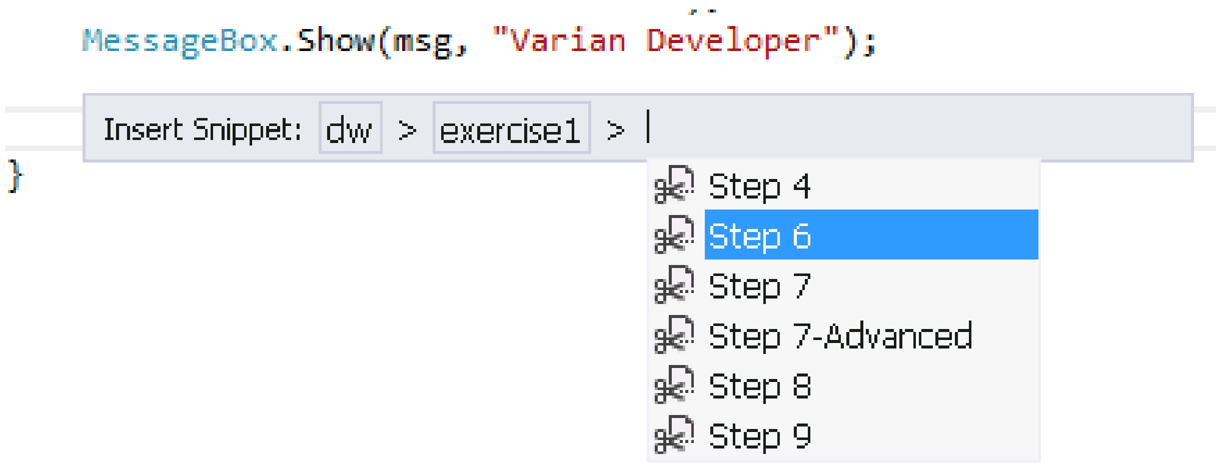


```
public IEnumerable<Structure> Structures { get; }
```

Exercise 1, Step 6

- Step 6: Select in code file, then right click to Insert Snippet…, choose dw -> exercise1 -> Step 6.





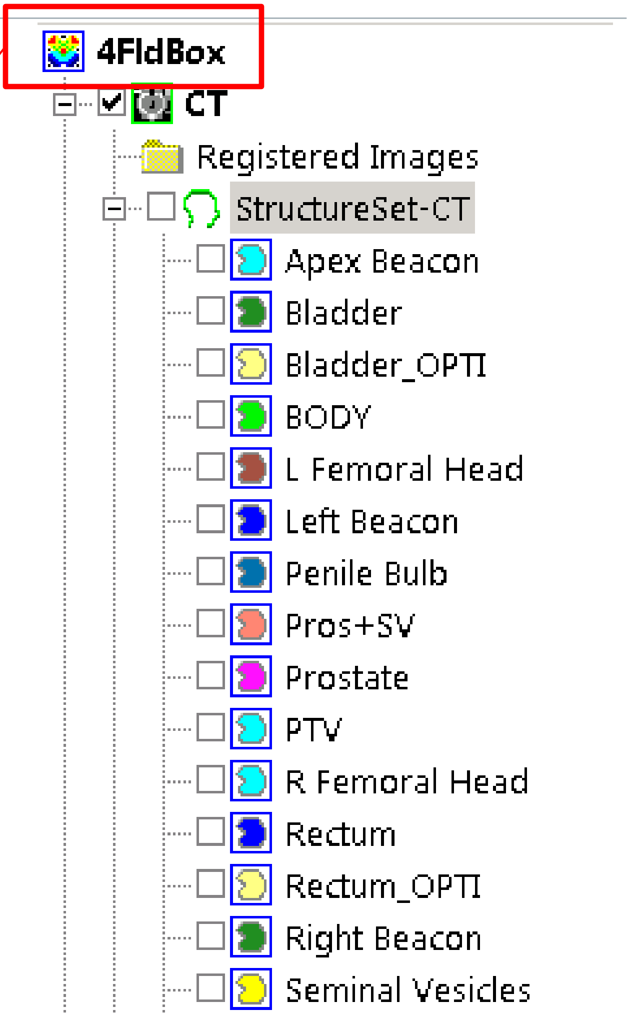
```
MessageBox.Show(msg, "Varian Developer");
```

Insert Snippet: dw > exercise1 >

```
}
```

Eclipse Context Notes

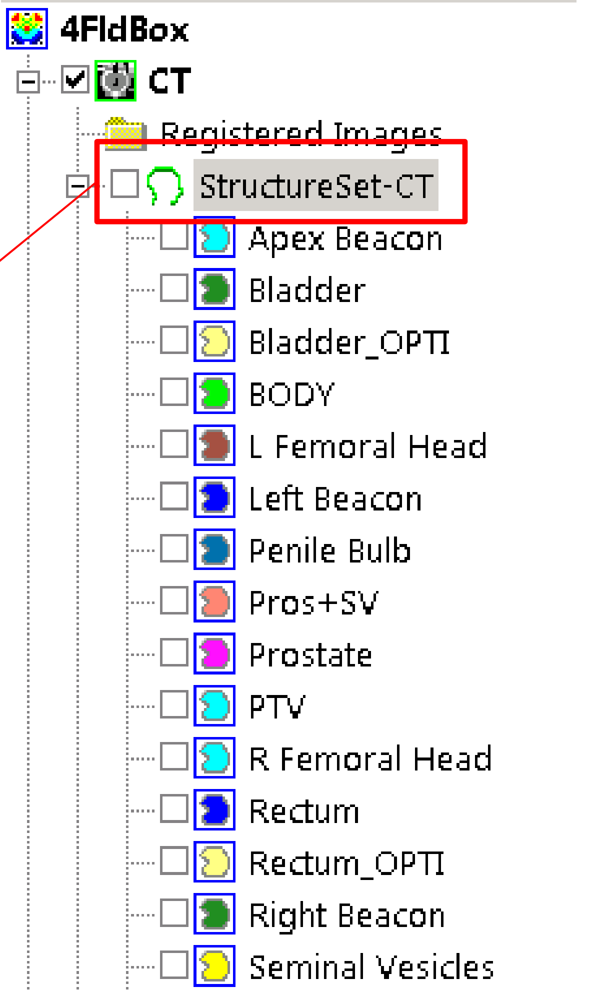
```
 // declare local variables
PlanSetup plan = context.PlanSetup;
StructureSet ss = context.StructureSet;
var listStructures = ss.Structures;
```



4FldBox
CT
Registered Images
StructureSet-CT
Apex Beacon
Bladder
Bladder_OPTI
BODY
L Femoral Head
Left Beacon
Penile Bulb
Pros+SV
Prostate
PTV
R Femoral Head
Rectum
Rectum_OPTI
Right Beacon
Seminal Vesicles

Eclipse Context Notes

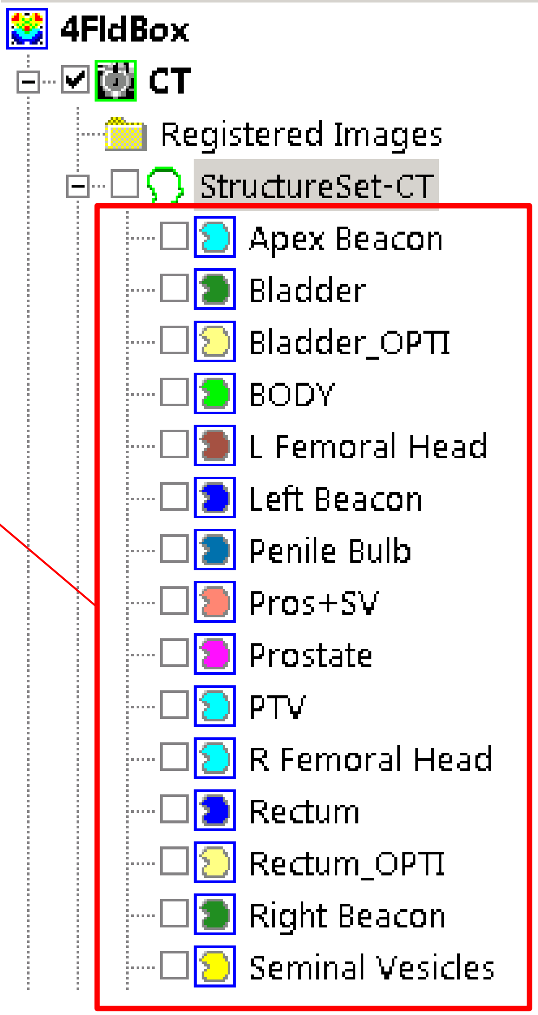
```
 // declare local variables
PlanSetup plan = context.PlanSetup;
StructureSet ss = context.StructureSet;
var listStructures = ss.Structures;
```



4FldBox
CT
Registered Images
StructureSet-CT
Apex Beacon
Bladder
Bladder_OPTI
BODY
L Femoral Head
Left Beacon
Penile Bulb
Pros+SV
Prostate
PTV
R Femoral Head
Rectum
Rectum_OPTI
Right Beacon
Seminal Vesicles

Eclipse Context Notes

```
 // declare local variables
PlanSetup plan = context.PlanSetup;
StructureSet ss = context.StructureSet;
var listStructures = ss.Structures;
```



4FldBox
CT
Registered Images
StructureSet-CT
Apex Beacon
Bladder
Bladder_OPTI
BODY
L Femoral Head
Left Beacon
Penile Bulb
Pros+SV
Prostate
PTV
R Femoral Head
Rectum
Rectum_OPTI
Right Beacon
Seminal Vesicles

C# Syntax – Arrays, Lists, Collections

```
// declare array of strings
string[] products = { "Eclipse",
"Truebeam", "Aria" };
// loop over array of strings
foreach (string product in products)
{
// do something
}
```

C# Lists and Collections

IEnumerable<T> Interface

(Enumerable of generic type 'T').

Enumerable – object you can loop over.

Just use 'var', and 'foreach'.

```
   foreach(Type name in IEnumerable<Type>)
```

IEnumerable<Structure>

(Enumerable of type 'Structure').

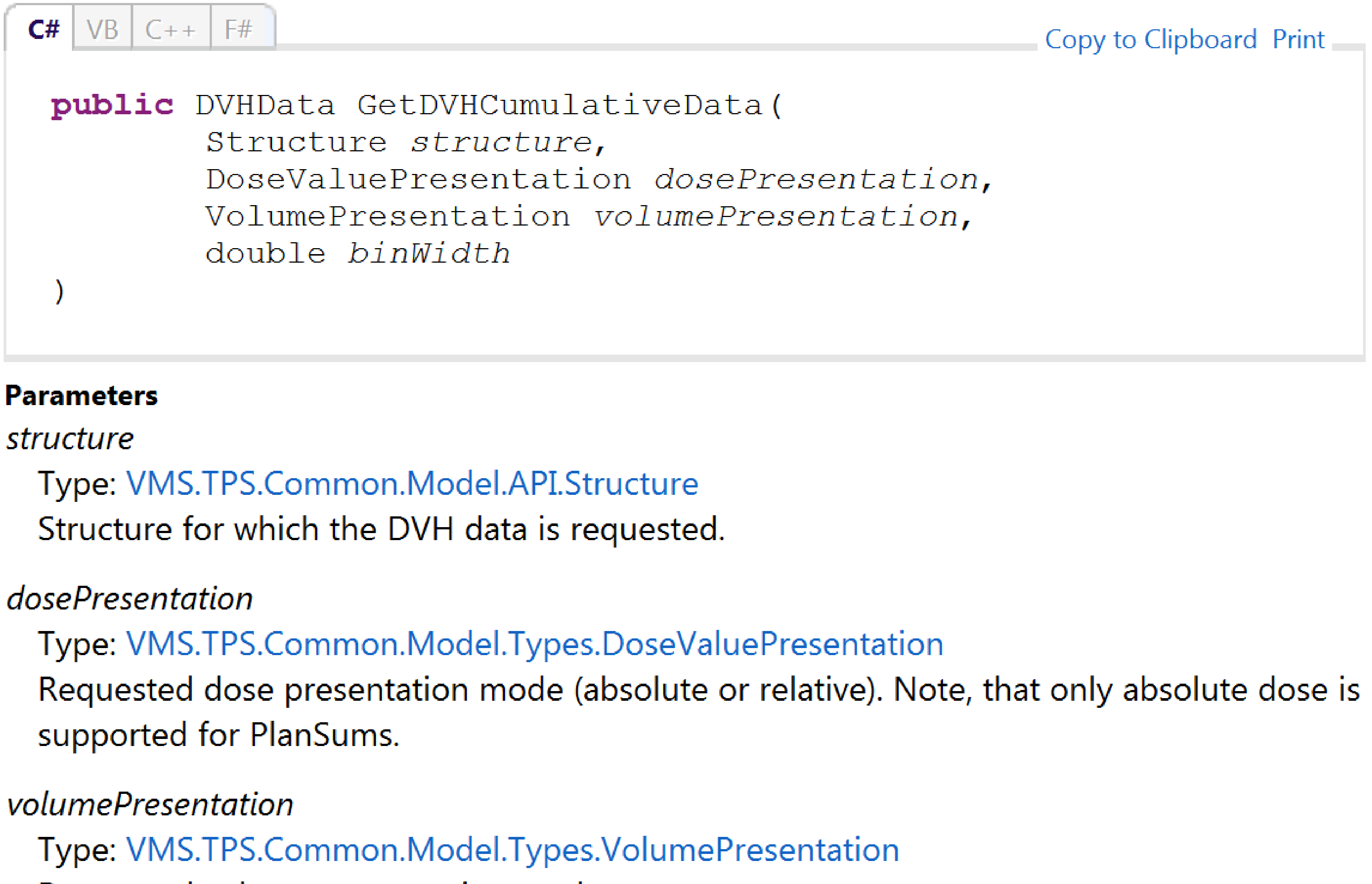
```
   StructureSet ss = context.StructureSet;
   foreach(Structure s in ss.Structures)
```

Exercise 1, Step 7

- Step 7: Write the code to loop over the list StructureSet.Structures, find the structure whose Id is “PTV”, and show its volume to the user.
- Compile and run the script in Eclipse.

[See code snippet for step 7 if you need help.]

Extracting DVHs…



```
public DVHData GetDVHCumulativeData(
	Structure structure,
	DoseValuePresentation dosePresentation,
	VolumePresentation volumePresentation,
	double binWidth
)
```

Parameters

structure

Type: VMS.TPS.Common.Model.API.Structure

Structure for which the DVH data is requested.

dosePresentation

Type: VMS.TPS.Common.Model.Types.DoseValuePresentation

Requested dose presentation mode (absolute or relative). Note, that only absolute dose is supported for PlanSums.

volumePresentation

Type: VMS.TPS.Common.Model.Types.VolumePresentation

Exercise 1, Step 8

Add the code to extract DVH data for PTV.

[See code snippet for Step 8 if you need help.]

Writing to a file in C#

File reading / writing in namespace System.IO.

System.IO.StreamWriter : a good option for general text file writing.

```
System.IO.StreamWriter dvhFile = new
System.IO.StreamWriter(@"C:\temp\keranendvh.txt");
dvhFile.WriteLine(“one line” );
dvhFile.Close();
```

VARIAN medical systems

Exercise 1, Step 9

- Add the code to write the DVH data for PTV to a .csv file.
  - Write code to open the .csv file
  - foreach loop over ptvDVH.CurveData
  - Write dose,volume to the file.
  - string.Format helps!
  - Close the file.

[See snippet for Step 9 if you need help.]

Congratulations, Scripter!

www.variandeveloper.com