VARIAN SCRIPTING EXERCISE 3

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.
- Developer Studio^{TM} is trademarked by Microsoft.

Exercise 3 Learning Goals

We will:

1) Understand 3D coordinates in ESAPI
2) Get dose values to points in 3D space
3) Get dose values along a line defined by two points in 3D space.

   This is called a dose profile

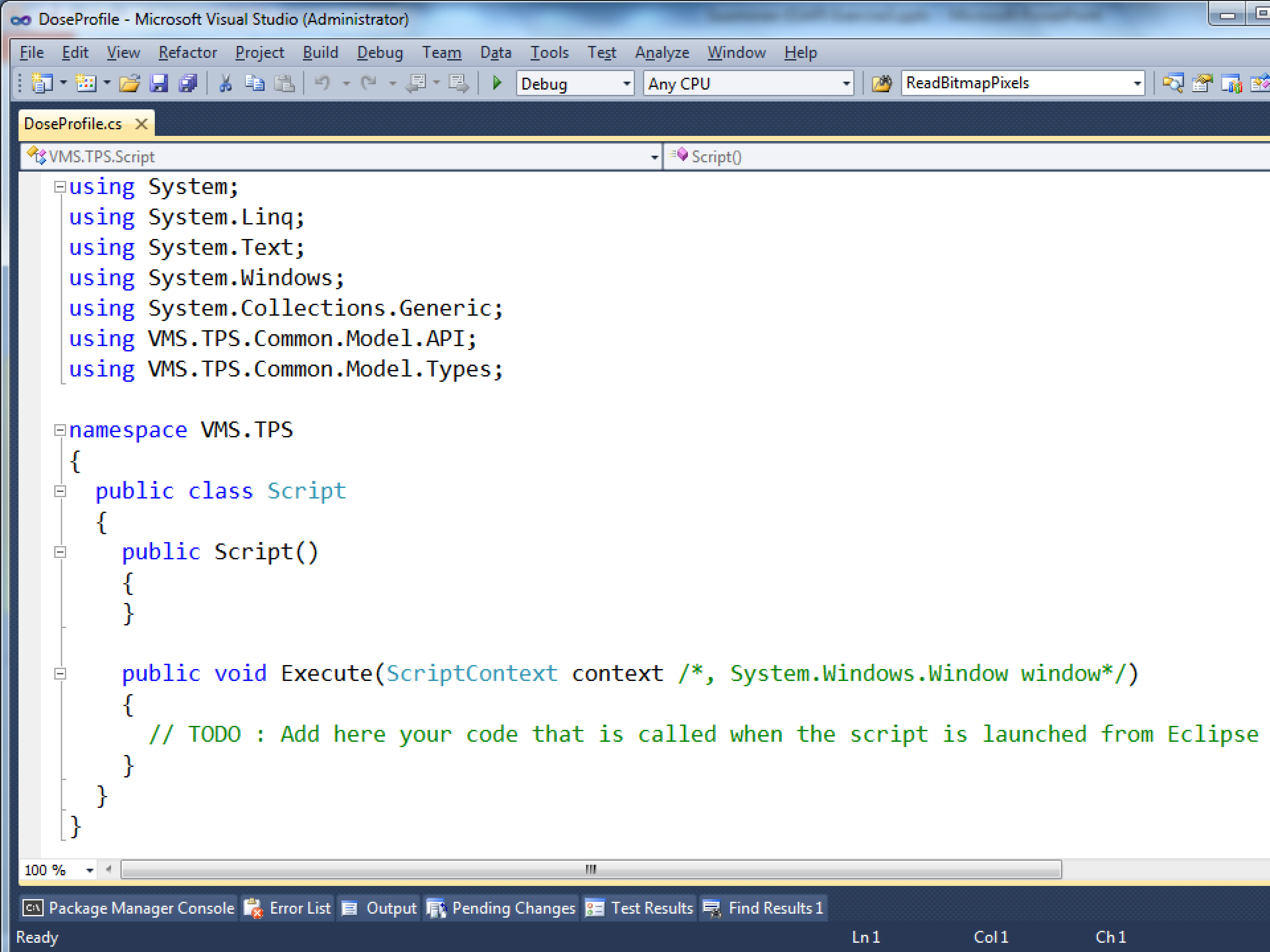
4) Export results to a file.

Sign in to Virtual Eclipse Environment

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

Exercise 3 – DoseProfiles

- Step 1: Run Script Wizard. Create a Single-file plug-in script and name it “DoseProfiles”, Open project in Developer Studio.

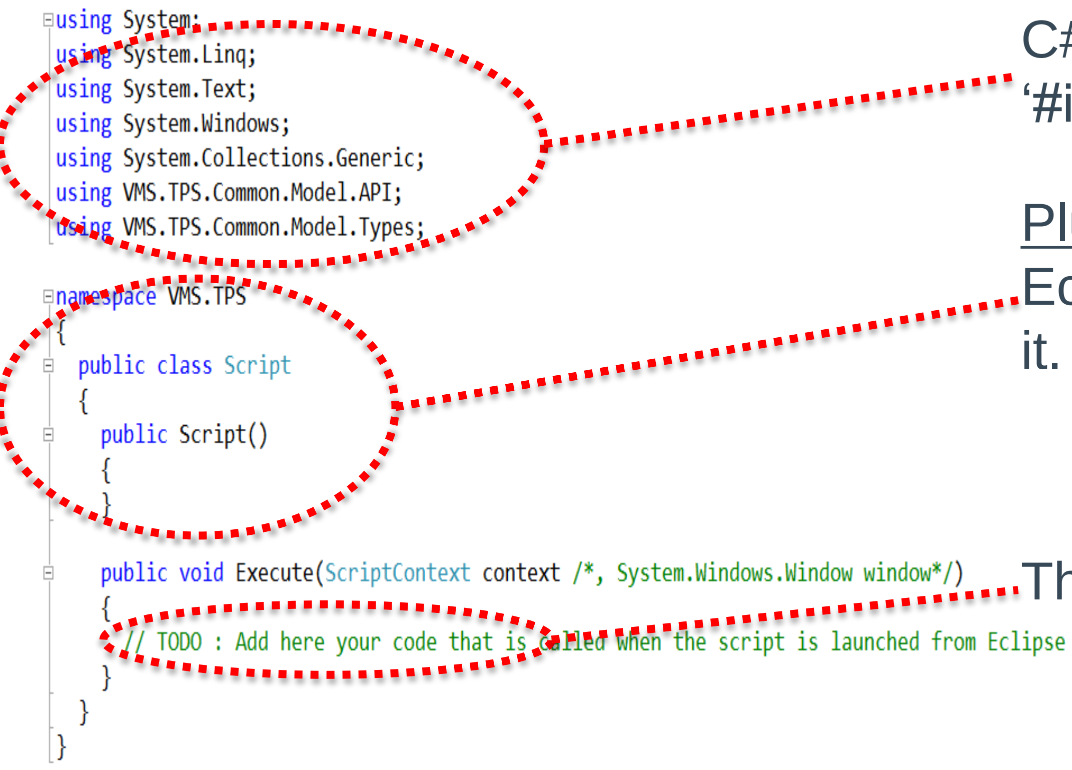


```
using System;
using System.Linq;
using System.Text;
using System.Windows;
using System.Collections.Generic;
using VMS.TPS.Common.Model.API;
using VMS.TPS.Common.Model.Types;

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 Eclipse
    }
  }
}
```

Plug-in script - C# Syntax Notes



```
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 Eclipse
    }
  }
}
```

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

Plug-in definitions lthis code helps Eclipse detect the plugin and load it.

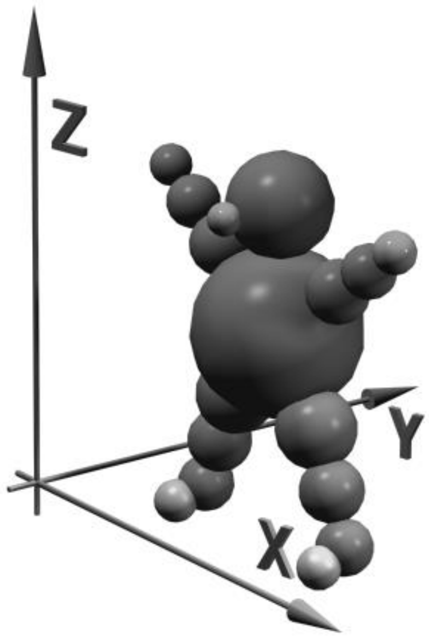
The real code starts here.

Plug-in Context

- For Plugin Script, Eclipse passes the context to script through variable ScriptContext.
- The context includes for example the patient, plan, structureset that are open in Eclipse at the moment of script launch.

Coordinates in 3D Space

- Coordinate system is DICOM
- Unit of measurement is millimeters
- The `VVector` object is used to represent positions in 3D space
- Note: Eclipse GUI uses the planning coordinate system. DicomToUser() and UserToDicom() conversion methods are available in the API.



Step 1: Isocenter Position of First Beam

- Use the following code to get and display the isocenter poisiton of first beam.

```
var isoc = context.PlanSetup.Beams.First().IsocenterPosition;

string msg = isoc.x.ToString() + ", " +
             isoc.y.ToString() + ", " +
             isoc.z.ToString();
MessageBox.Show("Isocenter (x,y,z in mm): " + msg);
```

- Open a plan and try it out (for example Eclipse-07, Varian, RA Phase1).

Relative and Absolute Doses

- As in Eclipse doses can be represented in relative (%) or Absolute units (Gy or cGy depending on your system configuration)
- Use `DoseValuePresentation` property of `PlanSetup` to select absolute or relative dose.

```
context.PlanSetup.DoseValuePresentation =
           DoseValuePresentation.Absolute;
```

VARIAN medical systems

Step 2: Get Dose at a Location

- Dose at any location can be obtained with `GetDoseToPoint()` method of a Dose object
- Add the following to get dose at isocenter:

```
var dose = context.PlanSetup.Dose.GetDoseToPoint(isoc);
MessageBox.Show("Dose at isocenter: " + dose.ToString());
```

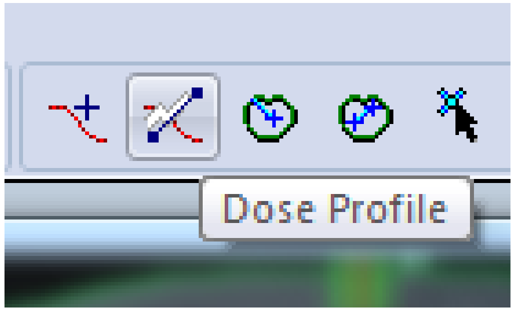
Dose Profiles

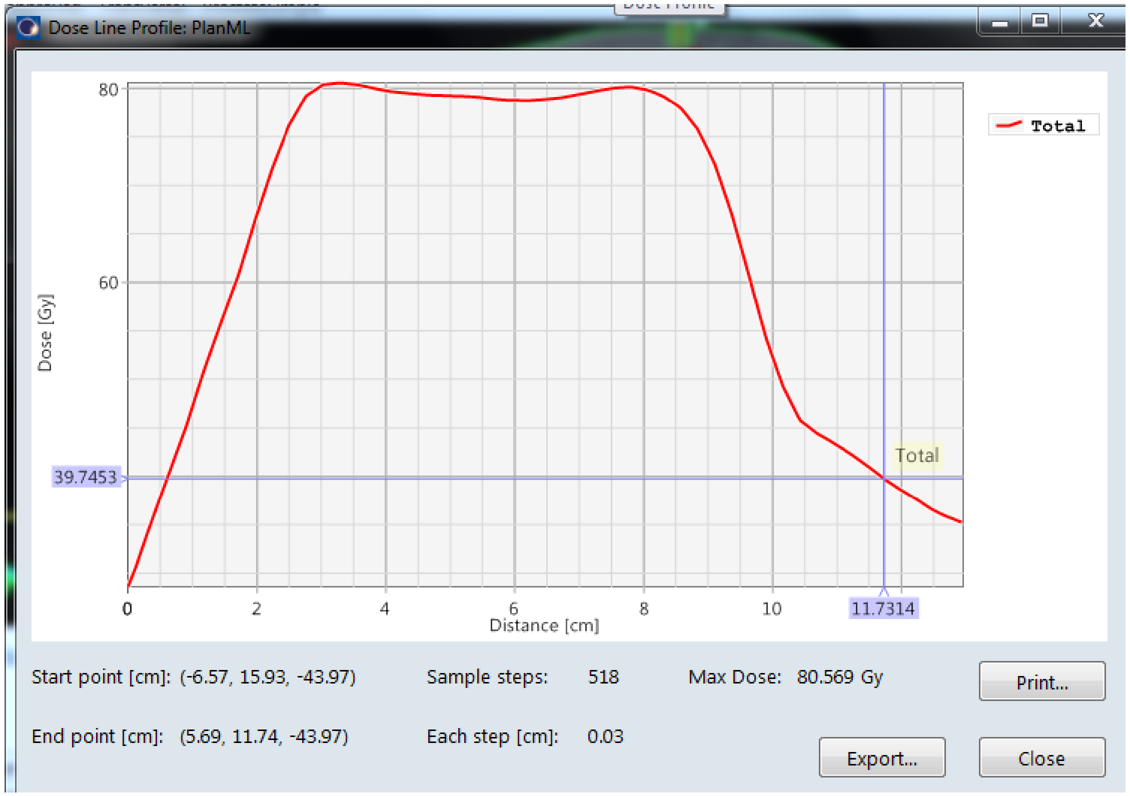
- Interpolated values along a line defined by two points in space are available with the `GetDoseProfile()` method of the `Dose` object.
- For better performance the buffer for the values is also given as input.

```
GetDoseProfile(VVector start, VVector stop, double[]
      preallocatedBuffer)
```

Dose Profiles continued

The concept of the profile is the same as in Eclipse





Start point [cm]: (-6.57, 15.93, -43.97)
End point [cm]: (5.69, 11.74, -43.97)
Sample steps: 518
Each step [cm]: 0.03
Max Dose: 80.569 Gy

Step 3: Dose Profile

Add the following lines to get 20 dose values between isocenter and location of dose maximum.

```
var dosemax = context.PlanSetup.Dose.DoseMax3DLocation;

double[] values = new double[20];
var profile = context.PlanSetup.Dose.GetDoseProfile(isoc,
                                                     dosemax,
                                                     values);
```

Step 3: Access the values of a profile

- The dose values of the profile are available as a list (an enumerable collection)
- Add the following lines to show the values to the user:

```
msg = String.Empty;
foreach (var profilePoint in profile)
{
  msg = msg + profilePoint.Value.ToString();
  msg = msg + "\n";
}
MessageBox.Show(msg);
```

Step 4: Writing Values to File

Use the following code to write the position and dose value to a comma separated file.

```
using (TextWriter writer = new StreamWriter("profile.txt"))
{
  writer.WriteLine("X, Y, Z, Dose");

  foreach (var profilePoint in profile)
  {
    writer.WriteLine(profilePoint.Position.x + "," +
                     profilePoint.Position.y + "," +
                     profilePoint.Position.z + "," +
                     profilePoint.Value);
  }
}
```

Other Available Profile Types

- Image profile gives HU values of points in the image
- Segment profile gives true/false values about points being inside/outside a structure.

Varian APIs – Enabling Innovation

www.variandeveloper.com