2015

Map editor

MOBILE DEVELOPMENT

JOEY DRIESSEN

Complete verandering van de code

De meeste gedeeltes van de code zijn aangepast volgens het OOP principe. Bijna alle functionaliteit is in klassen gestopt of zijn objecten voor gemaakt. Op deze manier is het overzichtelijker om aan te werken. Uiteraard was het ook een goeie oefening om te kijken of OOP ook echt duidelijk was voor mijzelf. Conclusie; nog veel werk aan de winkel op dat gebied.

Uitbreidende opties voor extra parameters

Er zitten verschillende opties in die meerdere parameters nodig hebben. Zo bijvoorbeeld een rechthoek tekenen. Om dit zichtbaar te maken voor de gebruiker zonder te veel onoverzichtelijke code wordt er gebruikt van bindingen.
Wanneer een optie is aangevinkt kijkt een "BooleanToVisibilityConverter" Of de optie is geselecteerd of niet. Op basis hier van wordt het stackpanel met de extra parameters zichtbaar.

Hieronder de code om dit werkend te krijgen.

Allereerst moet er een windows resource worden toegevoegd in XAML(MainWindow.XAML)..

```
<Window.Resources>
        <BooleanToVisibilityConverter x:Key="BooleanToVisibilityConverter"/>
    </Window.Resources>
```

Na dat dit gebeurd is worden er verschillende bindings gebruikt. Hier een voorbeeld voor de Random generator(MainWindow.XAML).:

```
<CheckBox Margin="0,0,0,5" Name="RandomMapCheckbox" Content="Random"/>
<StackPanel Visibility="{Binding ElementName=RandomMapCheckbox, Path=IsChecked,
Converter={StaticResource BooleanToVisibilityConverter}}" >
```

Dit concept meerdere malen terug in het programma vandaar hier uitgelegd.

Redo

De uitbreiding zorgt ervoor dat de gebruiker de acties die heeft ongedaan ook weer opnieuw kan maken. De functie werkt vanuit de edit menu of met CTRL+Y sneltoets.

Ik heb de functionaliteit toegevoegd om dit vanuit een menu te doen en ook vanuit een sneltoets.

De code is als volgt:

Als eerste moet er bij elke undo actie een redo object geplaatst worden in de redolist. Deze wordt hier RedoHistory genoemd(RedoUndo.cs - 59).

```
public void undoAction()
        {
            if (currentMap.UndoHistory.Count > 0)
            {

                RedoUndo lastaction = currentMap.UndoHistory.Pop();
                currentMap.RedoHistory.Push(lastaction);
                currentMap.CurrentMap.SetElement(lastaction.X, lastaction.Y,
lastaction.OriginalValue);
                if (currentMap.QueueList.Count > 0 && currentMap.QueueChecked)
                {
                    currentMap.QueueList.Dequeue();
                }
                currentMap.RenderMap();
            }
        }
```

Als er dan op redo geklikt wordt zal de volgende code worden uitgevoerd worden(RedoUndo.cs - 48):

```
public void redoAction()
        {
            if (currentMap.RedoHistory.Count > 0)
            {
                RedoUndo redoAction = currentMap.RedoHistory.Pop();
                currentMap.UndoHistory.Push(redoAction);
                currentMap.CurrentMap.SetElement(redoAction.X, redoAction.Y,
redoAction.typeBlock);
                currentMap.RenderMap();
            }
        }
```

Al de bovenstaande code kan gevonden worden in de “RedoUndo” klasse.
De volgende code roept de functie aan:

In mijn XAML staat het volgende(MainWindow.XAML).:

```
<MenuItem Header="Redo" Name="menuRedo" Click="menuRedo_Click"></MenuItem>
```

Waarbij dan het click event de volgende code bevat(MainWindow.XAML.cs - 126).:

```
private void menuRedo_Click(object sender, RoutedEventArgs e)
        {
            undomodel.redoAction();
        }
```

Voor de sneltoetsen wordt het volgende gedaan(MainWindow.XAML.cs - 280).:

```
private void Window_KeyDown(object sender, KeyEventArgs e)
        {

            if (e.KeyboardDevice.IsKeyDown(Key.LeftCtrl) || e.KeyboardDevice.IsKey-
Down(Key.RightCtrl))
            {
                switch (e.Key)
                {
                    case Key.Z:
                        undomodel.undoAction();
                        break;
                    case Key.Y:
                        undomodel.redoAction();
                        break;
                }

            }
        }
```

Show/hide Toolbox

De functionaliteit zorgt ervoor dat de toolbox kan verdwijnen zodat de gebruiker een groter veld heeft om zijn bewerking te doen. De toolbox kan makkelijk weg worden gedaan door de optie on de View knop.

De XAML code(MainWindow.XAML).:

```
<MenuItem Header="View" Name="menuView" Click="menuView_Click">
                <MenuItem Header="Hide Toolbox" Name="hideToolbox"
Click="hideToolbox_Click"></MenuItem>
            </MenuItem>
```

De C# code om het te laten verdwijnen of te laten zien(MainWindow.XAML.cs - 255).:

```
private void hideToolbox_Click(object sender, RoutedEventArgs e)
        {
            if (collapsed)
            {

                Toolbox.Visibility = Visibility.Visible;


                rowToolbox.Width = new GridLength(1, GridUnitType.Star);
                collapsed = false;
            }
            else
            {
                rowToolbox.Width = GridLength.Auto;
                Toolbox.Visibility = Visibility.Collapsed;
                collapsed = true;
            }
```

MessageBox'n

Er wordt bij het afsluiten of wanneer er een nieuwe map wordt gemaakt en er al een bestaat een melding gegeven. Dit geeft de gebruiker de kans om zijn huidige map eerste opslaan voordat er wordt afgesloten of een nieuwe map wordt gemaakt. Tevens kan de gebruiker ook de huidige map verwerpen of cancelen om te stoppen met afsluiten of een nieuwe map maken.

De code wordt op 2 verschillende plekken aangeroepen.

Wanneer de window wordt gesloten(MainWindow.XAML.cs - 315):

```
private void Window_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
            if (map.MapExists)
            {
                Warning warning = new Warning("Er is een actieve map wilt u deze eerste
opslaan?", "Opslaan huidige map", MessageBoxButton.YesNoCancel, MessageBoxImage.Warning);

                warning.CurrentMap = map;
                e.Cancel = warning.SaveCloseAppWarning();


            }

        }
```

Wanneer er een nieuwe map wordt gemaakt(MainWindow.XAML.cs - 53).

```
private void menuNew_Click(object sender, RoutedEventArgs e)
        {
            if (map.MapExists)
            {
                Warning warning = new Warning("Er is een actieve map wilt u deze eerste
opslaan?", "Opslaan huidige map", MessageBoxButton.YesNoCancel, MessageBoxImage.Warning);
                warning.CurrentMap = map;
                warning.SaveNewMapWarning();

            }
            else
            {
                MapDimensions askdims = new MapDimensions();
                askdims.ShowDialog();
                map.CurrentMap = new MapModel(askdims.Breedte, askdims.Hoogte);
                map.MapCanvas.Dispatcher.BeginInvoke(System.Windows.Threading.Dispatcher-
Priority.Background, (ThreadStart)delegate
                {
                    map.RenderMap();
                });

            }


        }
```

De save new map functionaliteit (Warning.cs – 75)

```
public void SaveNewMapWarning()
        {
            if (result == MessageBoxResult.Yes)
            {
                currentMap.SaveMap();
                MapDimensions askdims = new MapDimensions();
                askdims.ShowDialog();
                currentMap.CurrentMap = new MapModel(askdims.Breedte, askdims.Hoogte);

                currentMap.RenderMap();
            }
            if (result == MessageBoxResult.No)
            {

                MapDimensions askdims = new MapDimensions();
                askdims.ShowDialog();
                currentMap.CurrentMap = new MapModel(askdims.Breedte, askdims.Hoogte);

                currentMap.RenderMap();

            }
            if (result == MessageBoxResult.Cancel)
            {

            }
        }
```

De save close map functionaliteit(Warning.cs - 102)

```
public bool SaveCloseAppWarning()
        {
            if (result == MessageBoxResult.Yes)
            {
                currentMap.SaveMap();
                currentMap.CurrentMapPath = "";
                return false;
            }
            if (result == MessageBoxResult.No)
            {
                return false;
            }
            if (result == MessageBoxResult.Cancel)
            {
                return true;
            }
            else
            {
                return false;
            }
        }
```

Rechthoek tekenen

Deze functionaliteit dat de gebruiker een grote vierkant of rechthoek kan tekenen dan normaal. Er hoeft alleen een breedte en hoogte gegeven worden en dan wordt deze getekend wanneer er op het scherm wordt gedrukt.

In de volgende code wordt het aangeroepen tijdens het leftmousebuttonup (MainWindow.xaml.cs-136) event(MainWindow.xaml.cs - 155):

```
if (DrawRectangleCheckbox.IsChecked == true)
                {
                    customRectangle rectangle = new customRectangle();
                    rectangle.X = x;
                    rectangle.Y = y;
                    try
                    {
                        rectangle.Width = Convert.ToInt32(newRectangleBreedte.Text);
                        rectangle.Height = Convert.ToInt32(newRectangleHoogte.Text);
                    }
                    catch (FormatException)
                    {

                        MessageBox.Show("Alleen nummers mogen ingevuld worden");
                        newRectangleBreedte.Text =
newRectangleBreedte.placeHolderText;

                        newRectangleHoogte.Text = newRectangleHoogte.placeHolderText;
                    }
                    catch (Exception excp)
                    {
                        MessageBox.Show(excp.ToString());
                    }

                    rectangle.TypeBlock = Convert.ToInt32(t);

                    map.DrawOnMap(rectangle, click);
                }
```

Er wordt hier een customRectangle gemaakt en deze heeft de volgende properties(customRectangle.cs - 11):

```
private int x;
        private int y;
        private int typeBlock;
        private int width;
        private int height;

        public int Height
        {
            get { return height; }
            set { height = value; }
        }


        public int Width
        {
            get { return width; }
            set { width = value; }
        }

        public int TypeBlock
        {
            get { return typeBlock; }
            set { typeBlock = value; }
        }


        public int Y
        {
            get { return y; }
            set { y = value; }
        }

        public int X
        {
            get { return x; }
            set { x = value; }
        }
```

Er wordt hierna een draw functie aangeroepen. Deze is het zelfde als de andere draw functies echter accepteert deze een rectangle inplaats van een blok object.

De code ziet er als volgt uit(Map.cs - 258):

```
public void DrawOnMap(customRectangle rectangle, Point click)
        {
            int defaultY = rectangle.Y;
            if (QueueChecked)
            {
                for (int x = 0; x < rectangle.Width; x++)
                {
                    for (int y = 0; y < rectangle.Height; y++)
                    {
                        RedoUndo newAction = new RedoUndo(rectangle, this);
                        undoHistory.Push(newAction);
                        Queue tempQueue = new Queue(rectangle.X, rectangle.Y, Con-
vert.ToInt32(rectangle.TypeBlock), this);
                        tempQueue.QueueTask();
                        rectangle.Y += 1;
                    }
                    rectangle.Y = defaultY;
                    rectangle.X += 1;
                }

            }
            else
            {
                for (int x = 0; x < rectangle.Width; x++)
                {
                    for (int y = 0; y < rectangle.Height; y++)
                    {
                        RedoUndo newAction = new RedoUndo(rectangle, this);
                        undoHistory.Push(newAction);
                        currentMap.SetElement(rectangle.X, rectangle.Y, Con-
vert.ToInt32(rectangle.TypeBlock));
                        RenderMap();
                        rectangle.Y += 1;
                    }
                    rectangle.Y = defaultY;
                    rectangle.X += 1;

                }


            }
        }
```

Klik en sleep

Deze functie klik en sleep aan voor de gebruiker. Dit houd in dat wanneer de gebruiker klikt en sleept er meerdere blokken kunnen worden getekend (ook vierkanten/rechthoeken zoals uitgelegd in de vorige functie). Helaas zijn hier nog wel enige performance issues die grotendeels worden opgelost met de queue tasks knop maar helaas niet met live editing tot dusver.

De functie wordt hier aangeroepen en is tevens ook de functionaliteit(MainWindow.xaml.cs - 195):

```
private void mapCanvas_MouseMove(object sender, MouseEventArgs e)
        {
            if (cmbBrush.SelectedIndex > -1 && e.LeftButton == MouseButtonState.Pres-
sed)
            {
                if (queueCheckbox.IsChecked == true)
                {
                    map.QueueChecked = true;
                }
                else
                {
                    map.QueueChecked = false;
                }
                Point click = e.MouseDevice.GetPosition(mapCanvas);
                string t = (cmbBrush.SelectedItem as ComboBoxItem).Content.ToString();

                int x = (int)((click.X / map.BlockScale));
                int y = (int)((click.Y / map.BlockScale));

                if (DrawRectangleCheckbox.IsChecked == true)
                {
                    customRectangle rectangle = new customRectangle();
                    rectangle.X = x;
                    rectangle.Y = y;
                    try
                    {
                        rectangle.Width = Convert.ToInt32(newRectangleBreedte.Text);
                        rectangle.Height = Convert.ToInt32(newRectangleHoogte.Text);
                    }
                    catch (FormatException)
                    {

                        MessageBox.Show("Alleen nummers mogen ingevuld worden");
                        newRectangleBreedte.Text = newRectangleBreedte.placeHolder-
Text;


                        newRectangleHoogte.Text = newRectangleHoogte.placeHolderText;
                    }
                    catch (Exception excp)
                    {
                        MessageBox.Show(excp.ToString());
                    }

                    rectangle.TypeBlock = Convert.ToInt32(t);

                    map.DrawOnMap(rectangle, click);
                }
                else
                {
                    singleBlock.X = x;
                    singleBlock.Y = y;
                    singleBlock.TypeBlock = Convert.ToInt32(t);

                    map.DrawOnMap(singleBlock, click);
                }



            }
        }
```

Random map/Random map pro

De gebruiker kan via de toolbox zijn map random laten invullen. Hier zijn verschillende opties voor.

De gebruiker kan kiezen hoeveel het minimum aantal blokjes van een gekozen type in de map moeten komen. De map kan de huidige map zijn of een nieuwe map waar de gebruiker de grote van de map kan meegeven.
Mocht de onderstaande XAML code niet echt leesbaar zijn het effect is beter te zien wanneer het project gerund wordt.

De code wordt als volgt aangeroepen(MainWindow.xaml):

```
<CheckBox Margin="0,0,0,5" Name="RandomMapCheckbox" Content="Random"/>
                    <StackPanel Visibility="{Binding ElementName=RandomMapCheckbox,
Path=IsChecked, Converter={StaticResource BooleanToVisibilityConverter}}" >
                        <TextBlock Margin="0,0,0,5" Text="if left empty default values are
used" TextWrapping="Wrap"></TextBlock>
                        <Separator/>
                        <TextBlock Margin="0,0,0,5" Text="Minimum blocks with type" Tex-
tWrapping="Wrap"></TextBlock>
                        <local:CustomTextbox Margin="0,0,0,5"  x:Name="minTypeTextBox"
placeHolderText="Fill in Type(1,2,..)" />
                        <local:CustomTextbox Margin="0,0,0,5"  x:Name="minAmountTextBox"
placeHolderText="Fill in amount"/>
                        <Separator></Separator>
                        <RadioButton Margin="0,0,0,5" GroupName="TypeMap" Content="Use
current map"  Name="UseCurrentMapRadioButton" IsChecked="True" Checked="TypeMapRadio-
Button_Checked"></RadioButton>
                        <RadioButton Margin="0,0,0,5" GroupName="TypeMap" Content="Create
new map" Name="CreateNewMapRadioButton" Checked="TypeMapRadioButton_Checked"></Radio-
Button>
                        <StackPanel Visibility="{Binding ElementName=CreateNewMapRadio-
Button, Path=IsChecked, Converter={StaticResource BooleanToVisibilityConverter}}">
                            <local:CustomTextbox x:Name="newMapBreedte" placeHolder-
Text="Width"></local:CustomTextbox>
                            <local:CustomTextbox x:Name="newMapHoogte" placeHolder-
Text="Height"></local:CustomTextbox>
                        </StackPanel>
                        <Button Name="CreateRandomMap" Content="Randomize" Click="Crea-
teRandomMap_Click"></Button>
                    </StackPanel>
```

De echte functie wordt aangeroepen vanuit de button en het event “CreateRandomMap_Click” wordt aangeroepen.
Dit is de functionaliteit die in het event zit(MainWindow.xaml.cs – 195):

```
private void CreateRandomMap_Click(object sender, RoutedEventArgs e)
        {
            if (map.CreateNewMap)
            {

                try
                {
                    map.NewMapBreedte = Convert.ToInt32(newMapBreedte.Text);
                    map.NewMapHoogte = Convert.ToInt32(newMapHoogte.Text);
                }
                catch (FormatException)
                {

                    MessageBox.Show("Alleen nummers mogen ingevuld worden");
                    newMapBreedte.Text = newMapBreedte.placeHolderText;

                    newMapHoogte.Text = newMapHoogte.placeHolderText;
                }
                catch (Exception excp)
                {
                    MessageBox.Show(excp.ToString());
                }

            }

            int parseValue;
            if (!int.TryParse(minTypeTextBox.Text, out parseValue) && !int.Try-
Parse(minTypeTextBox.Text, out parseValue))

            {
                //the input is not a number. This could be the helper text or a wrong in-
put
                map.RenderRandomMap();
            }
            else
            {
                int type = Convert.ToInt32(minTypeTextBox.Text);
                int amount = Convert.ToInt32(minAmountTextBox.Text);
                map.RenderRandomMap(type, amount);
            }

        }
```

Hier onder de code voor een default random map te maken. Hier wordt geen rekening gehouden met een minimum aantal blokjes(Map.cs - 299):

```
public void RenderRandomMap()
        {
            Random r = new Random();
            if (createNewMap)
            {

                currentMap = new MapModel(NewMapBreedte, NewMapHoogte);
            }
            int type = r.Next(0, 4);
            for (int x = 0; x < currentMap.Breedte; x++)
            {
                for (int y = 0; y < currentMap.Hoogte; y++)
                {

                    currentMap.SetElement(x, y, type);
                    type = r.Next(0, 4);
                }
            }

            this.RenderMap();


        }
```

Hier onder de code waar wel rekening met minimum aantal blokjes wordt gehouden(Map.cs - 322):

```
public void RenderRandomMap(int typeBlock, int amount)
        {
            Random r = new Random();
            if (createNewMap)
            {

                currentMap = new MapModel(NewMapBreedte, NewMapHoogte);
            }
            int type = r.Next(0, 4);
            if (amount > (currentMap.Breedte * currentMap.Hoogte))
            {
                amount = currentMap.Breedte * currentMap.Hoogte;
            }
            while (amount > 0)
            {
                for (int x = 0; x < currentMap.Breedte; x++)
                {
                    for (int y = 0; y < currentMap.Hoogte; y++)
                    {
                        if (type == typeBlock && currentMap.GetElement(x, y) != type)
                        {
                            int current = currentMap.GetElement(x, y);
                            currentMap.SetElement(x, y, type);
                            amount--;



                        }
                        else
                        {
                            if (currentMap.GetElement(x, y) != typeBlock)
                            {
                                currentMap.SetElement(x, y, type);
                            }

                        }

                        type = r.Next(0, 4);

                    }
                }
            }


            this.RenderMap();
        }
```

Exception handling

Tijdens het development zijn niet veel excepties opgekomen. Degene die wel voorbij zijn gekomen zal ik hier onder beschrijven waar ze te vinden zijn. Sommige “excepties” worden in runtime opgelost met en if statement of omrekening van bijvoorbeeld een outofindex.

Verkeerde input grotere rechthoek/vierkant(MainWindows.xaml.cs - 160):

```
try
                    {
                        rectangle.Width = Convert.ToInt32(newRectangleBreedte.Text);
                        rectangle.Height = Convert.ToInt32(newRectangleHoogte.Text);
                    }
                    catch (FormatException)
                    {

                        MessageBox.Show("Alleen nummers mogen ingevuld worden");
                        newRectangleBreedte.Text = newRectangleBreedte.placeHolderText;

                        newRectangleHoogte.Text = newRectangleHoogte.placeHolderText;
                    }
                    catch (Exception excp)
                    {
                        MessageBox.Show(excp.ToString());
                    }
```

Verkeerde input nieuwe map(MainWindows.xaml.cs - 195):

```
try
                {
                    map.NewMapBreedte = Convert.ToInt32(newMapBreedte.Text);
                    map.NewMapHoogte = Convert.ToInt32(newMapHoogte.Text);
                }
                catch (FormatException)
                {

                    MessageBox.Show("Alleen nummers mogen ingevuld worden");
                    newMapBreedte.Text = newMapBreedte.placeHolderText;

                    newMapHoogte.Text = newMapHoogte.placeHolderText;
                }
                catch (Exception excp)
                {
                    MessageBox.Show(excp.ToString());
                }
```

Kleinere map maken dan mogelijk is(MapDimensions.xaml.cs - 30):

```
try
            {
                Hoogte = Convert.ToInt32(txbHoogte.Text);
                Breedte = Convert.ToInt32(txbBreedte.Text);
                this.Close();
            }
            catch (ArgumentOutOfRangeException excep)
            {
                MessageBox.Show(excep.Message);
            }
            catch (FormatException excep)
            {
                MessageBox.Show("vul alleen gehele getallen boven 0 in:\n"+excep.Message);
            }
```