# ThingBuilder

## Version 0.15

## October 2013

**Brian Farrimond**

# Contents

# Introduction

**ThingBuilder** creates dynamic 3D models of a range of things including sailing ships, paddle steamers and scenario scenes including harbours.

The models can be incorporated within 3D simulations of scenarios generated in the **ScenarioBuilder** tool.

Figure 1 shows a sailing ship model under development in ThingBuilder.
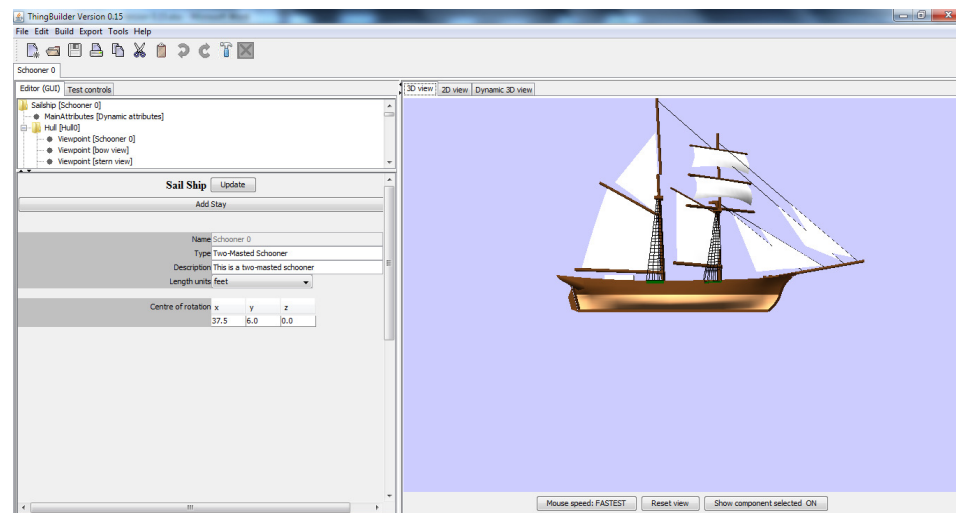


**Figure 1: Developing a 3D model of a sailing ship in ThingBuilder**

The dynamic aspects of a model can be tested within **ThingBuilder**. Figure 2 shows the sailing ship with sails furled. The dynamic components are controlled by the sliders in the left hand side panel.
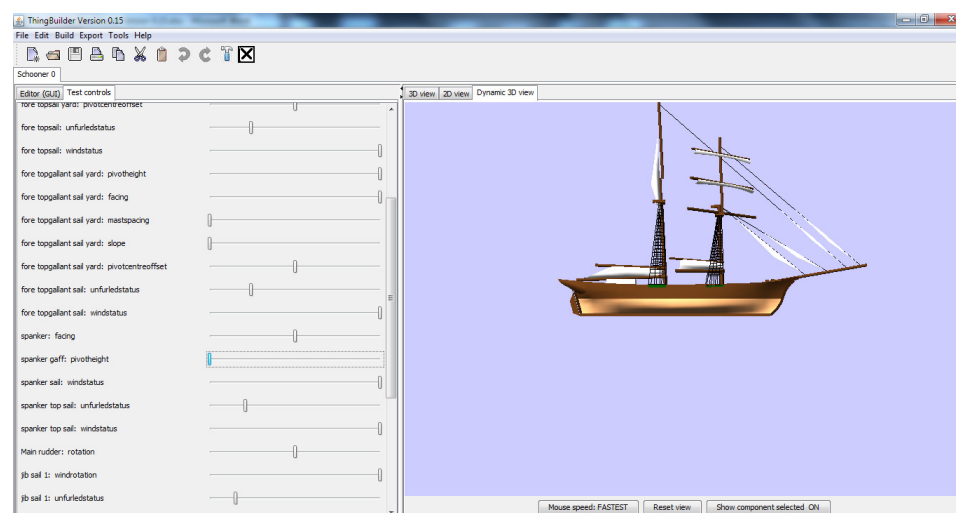


**Figure 2: Changing dynamic components - furling the sails**

When the model is used in a scenario, **ScenarioBuilder** controls the dynamic components.

Figure 3 shows a scenario scene under development
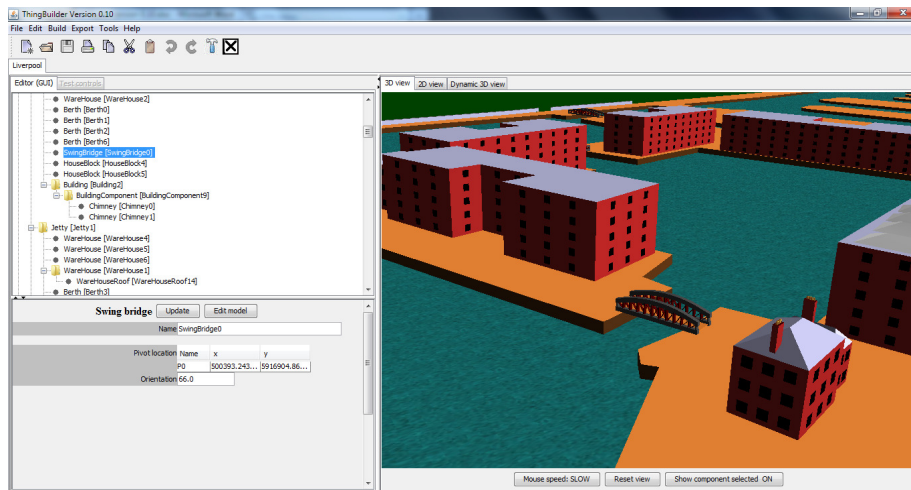


**Figure 3: Developing a model of a harbour in ThingBuilder**

Figure 4 shows a dynamic swing bridge components being tested.
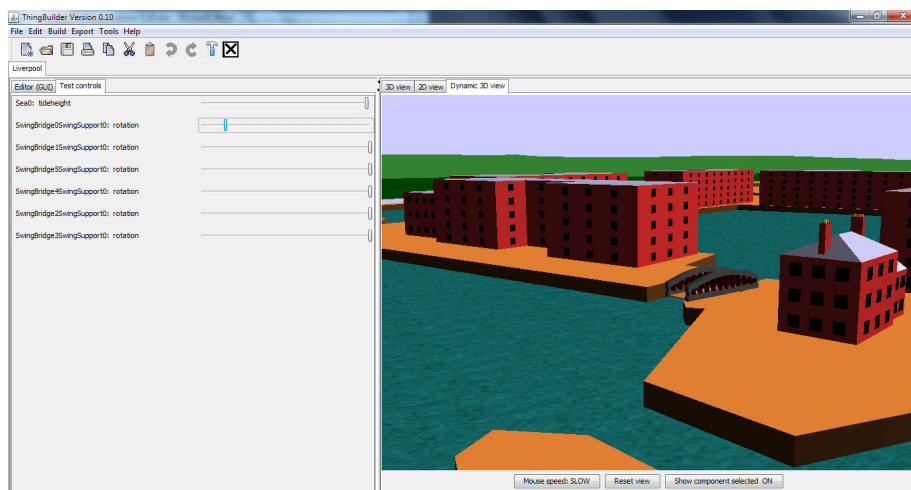


**Figure 4: Changing dynamic components - opening a swing bridge**

The models can be quite extensive as shown in Figure 5 below.
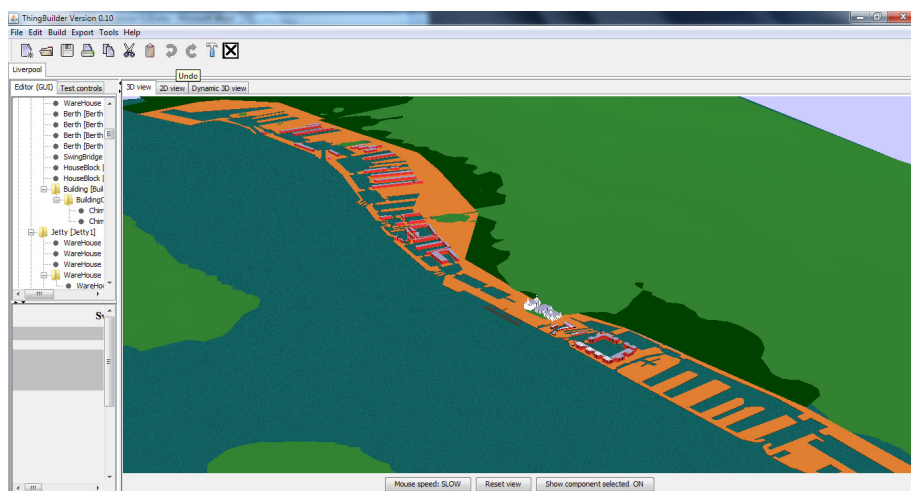


**Figure 5: Liverpool docks model under development**

# Project web site

The ThingBuilder application and this user guide can be found at:

http://www.edgehill.ac.uk/computing/3d-parametric-modelling/

The installation file, the Java runtime environment and the user guide for ThjingBuilder can be found on the Download page.

## Installing ThingBuilder

**ThingBuilder** is a Windows program and runs on Windows 2000, Windows XP, Windows 7. It is not yet tested on Windows 8. It is installed by using a setup file. The current setup file is *thingbuildersetup015.exe.*

Obtain a copy of the setup file and launch it by double clicking on its icon in Windows Explorer.

The default installation places **ThingBuilder** in the folder *c:\ThingBuilder*.

**ThingBuilder** is a Java 3D 32-bit application and so the 32-bit Java Runtime environment must be installed for it to run. The Java 3D libraries are included in the installation.

There are issues with Java 3D and Java 7 so we currently use a Java 6 JRE. **ThingBuilder** works successfully with the JRE installed by *jre-6u24-windows-i586.exe* . This file can be found on the downloads page of the Builder web site.

# Running ThingBuilder

Once installed, run **ThingBuilder** like any other installed program, through the Start menu or the Desktop shortcut that may be optionally installed.

When **ThingBuilder** starts, the screen shown below in Figure 6 is opened.



**Figure 6: ThingBuilder opening screen**

# ThingBuilder Menus

## *File*

| | |
|---|---|
| **New Thing** | Create a new thing (sailing ship, paddle steamer etc) |
| **Open Thing** | Open an existing thing. Several things can be open at the same time. |
| **Save** | Save the currently selected open thing |
| **Save As** | Save the currently selected open thing as a new thing |
| **Close** | Close the currently selected open thing |
| **Add To Subcomponent Library** | Things can be subcomponents of other things. e.g. swing bridges are subcomponents of harbours. This menu item saves the selected open thing to the subcomponent library from where it can be added to a parent thing. |
| **Add To Subcomponent Library As** | Save a thing to the subcomponent library under a new name. |
| **Print** | Not yet implemented. |

## *Edit*

| | |
|---|---|
| **Undo** | Undo the last editing action |
| **Redo** | Redo the last undone editing action. |
| **Cut** | Cut the currently selected component |
| **Copy** | Copy the currently selected component into the copy buffer. |
| **Paste** | Paste the contents of the copy buffer to the currently selected component. This only works if the copy buffer contents are a valid subcomponent of the currently selected component. |

## *Build*

| | |
|---|---|
| **Build Dynamic Model** | Generates the dynamic model and displays it in the right hand panel under the *Dynamic 3D view* tab.. |

## *Export*

| | |
|---|---|
| **Export Model to ScenarioBuilder** | Export the set of files comprising the dynamic model to the ScenarioBuilder resources database |

## *Tools*

| Options | Select the path to the installation of ScenarioBuilder. |
|---|---|
| **Register An Image** | Associate selected pixels on an image with physical location e.g. Universal Transverse Mercator (UTM) coordinates of points on a map of an area. UTM is an alternative to latitude and longitude in which the values of the coordinates are measured in meters instead of degrees, minutes and seconds. |
| **Edit A Registered Image** | Registered images may be edited or deleted. |

The menu item **Tools | Options** opens the dialog shown below in Figure 7:
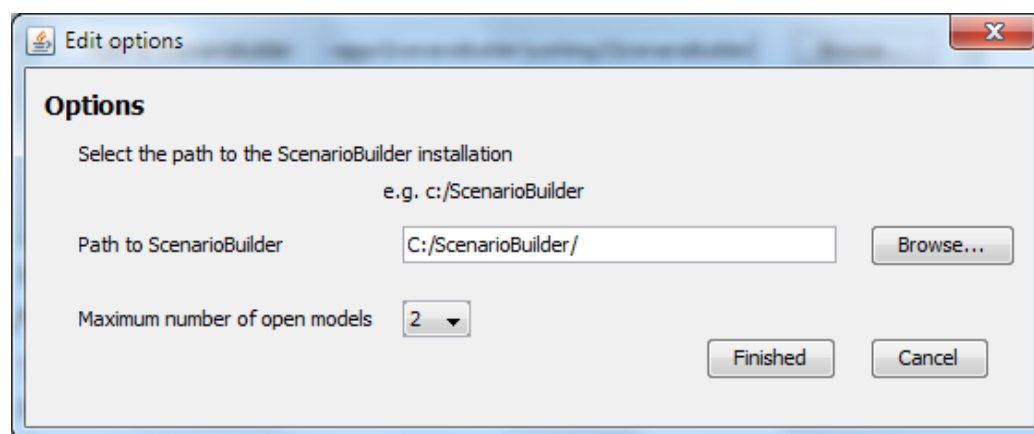


**Figure 7: The Options dialog**

*Path to ScenarioBuilder*      identifies the folders containing **ScenarioBuilder**. **ThingBuilder** uses this path to export models to ScenarioBuilder.

*Maximum number of open models*      determines how many models can be opened simultaneously in ThingBuilder. The issue is that opening too many models causes out of memory errors. The number of models that can be opened depends on memory size and size of the models.

Registering and editing images are described in the next section.

# Registering an image

Images from sources such as maps and Google Earth can be used to create geographically accurate models of, for example, docks, streets, churches etc. Certain components within ThingBuilder are added by adding points to a 2D image e.g. the outline of a building or a dock floor plan. The 2D screen can be enhanced with scaled backdrops of map and photo images to enable accurate plotting of points.

The images need to be scaled correctly to ensure accurate plotting. This is achieved by **registering** the images.

Register an image by doing the following:

1. Place the image in any suitable folder
2. Select the menu item **Tools | Register An Image...**

This opens the dialog shown below in Figure 8 .



**Figure 8: Registering an image**

Follow the steps shown on the right hand side:

| Step 1  Select an image. | A separate copy of the image will be made and placed inside the ThingBuilder installation directories |
|---|---|
| Step 2 Register the image | Identify two points on the Google Earth equivalent of your image and write down the UTM coordinates of the two points. Google Earth displays the coordinates of the cursor in latitude and longitude by default at the bottom centre of the window. You can change this to UTM by selecting the Google Earth  menu item **Tools | Options**. Select the **3D View** tab and then the **Univeral Transverse Mercator** radio button as shown below in Figure 9.  Finally, click on OK. |
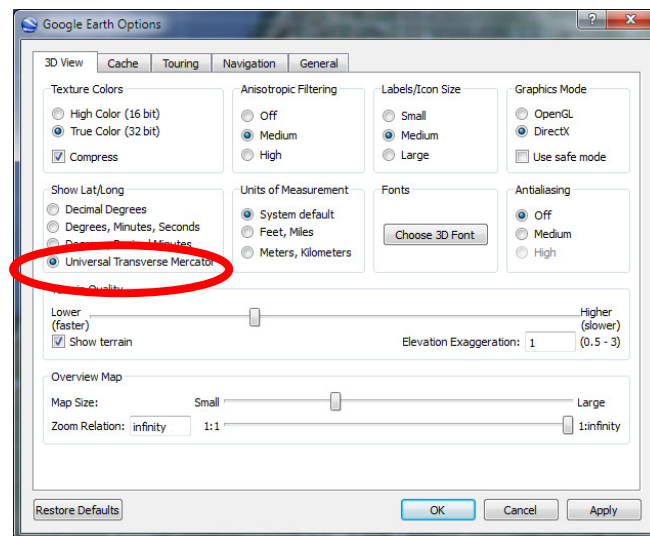


**Figure 9: Google Earth Options**

Now you will see at the bottom of the Google Earth window something like Figure 10 below.



**Figure 10: UTM coordinates in Google Earth**

**30 U** refers to a latitude band within one of the 60 zones into which the earth's surface is sub-divided. Each zone is 6 degrees of longitude wide. Each zone is split into 20 latitude bands. Thus 30 U refers to band U within zone 30.

**487144.04 m E** is known as an "easting" and refers to the cursor being at a point that many metres east of an arbitrary line through the zone.

**5930308.15 m N** is known as a "northing" and refers to the cursor being at a point that many metres north of an arbitrary line through the zone.

**elev -5 m** refers to the elevation in metres of the point above sea level.  GoogleEarth currrently uses the SRTM elevation data for its terrain. This is the same

data that ThingBuilder uses for terrain. SRTM data is described below.

The wider apart both in north-south and east-west directions the points are the more accurate the registration. Click on each point in the image and enter the UTM coordinates you have written down for each point. More on this below.

Step 3 Fill in the paperwork     Fill in the fields identifying the image name and label for the image to appear in ThingBuilder along with description and file type to use.

Step 4 Save the registration     This will add the image along with a record of the physical equivalents of the two selected pixels to ThingBuilder's registered image library.

## *Good (and bad) registration*

Points will be rounded off to the nearest whole pixel and so greatest accuracy will be achieved if the points are vertically and horizontally as far apart as possible. Figure 11 below shows poor and good pairs of points.



Poor         Poor         Good
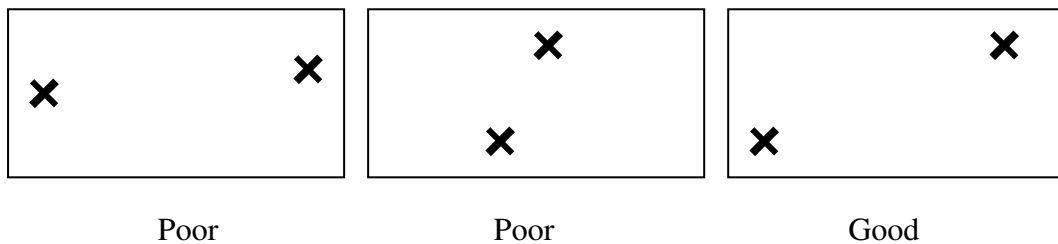
**Figure 11: Bad and good registration points**

The first one is poor because the difference in height of the two selected points is small. Thus a 1 pixel rounding of the points in the image may cause a 10% error if the height difference is 10 pixels. If the difference in height is 100 pixels then the error is only 1%.

Similarly the second example is poor because the horizontal difference is too small.

The thirs one is good because both horizontal and vertical deifferences are big - thus both errors will be small.

# Editing a registered image

A registered image may be edited or deleted from the library of registered images by selecting the menu item **Tools | Edit A Registered Image...** .

The dialog shown below in  appears;
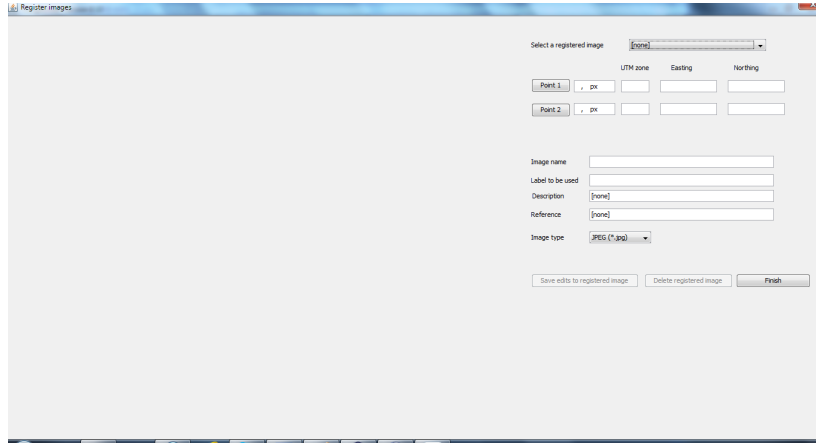


**Figure 12: Editing or deleting a registered image**

# Creating a new thing

To start to create a new thing, click on the menu item **File | New Thing...** This opens the New Tjhing Wizard shown below in Figure 13
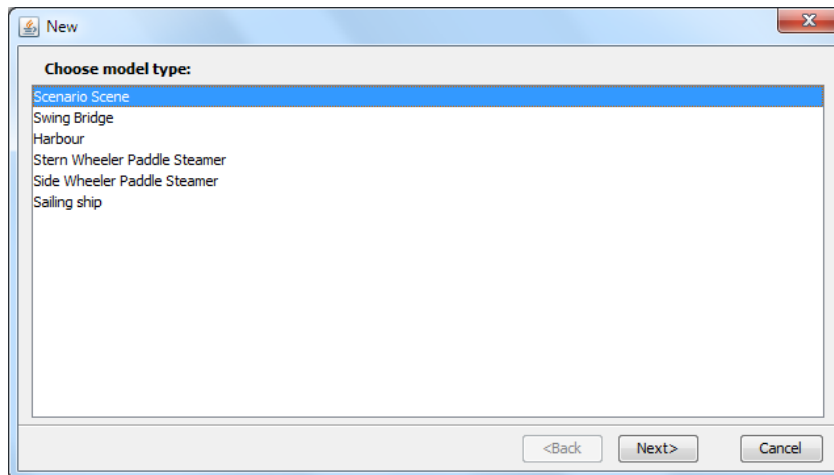


**Figure 13: The New Thing Wizard**

Select a type of thing from the list and click on **Next**.

The next page of the Wizard opens as shown below in Figure 14:
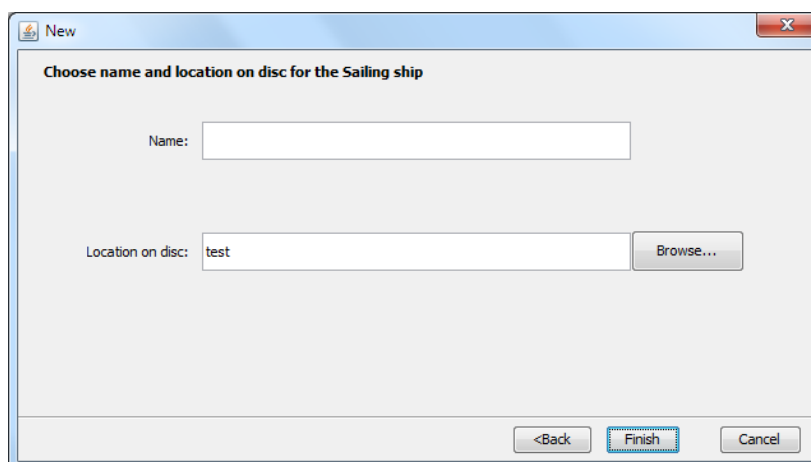


**Figure 14: Second page on the New Thing Wizard**

The **Name** and **Location on disc** fields are used to identify the thing and locate it on disc.

Click on **Finish** to complete the creation of the new thing.

The beginnings of a new thing of the chosen type will be displayed in the right hand panel.

Here in Figure 1, for example, is the beginnings of a sailing ship.It has a hull and a bowsprit but no masts.
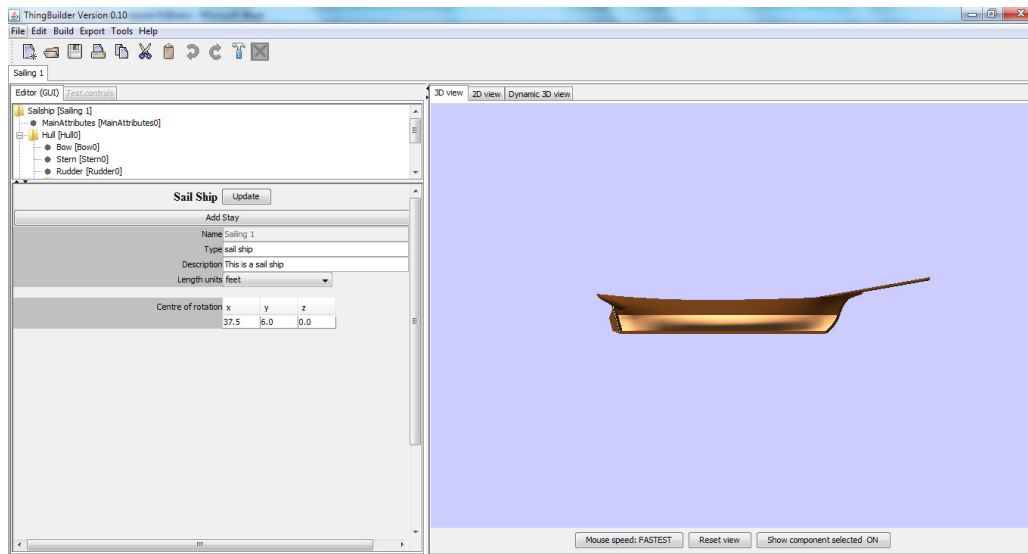


**Figure 15: Creating a new sailing ship**

The 3D view is in the right hand panel. The left hand panel is split into two sections.

- the top section is the **tree view** which shows the component "tree" for the model (shown below in  Figure 16 in more detail). This is where components of the model are selected for editing.
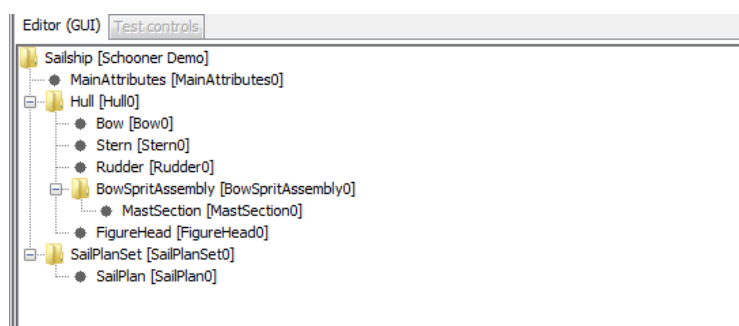


**Figure 16: The component tree**

When a component is selected in this tree, it turns blue in the 3D view

- the bottom section is the **attribute panel** which shows the controls for the currently selected component. When no component is selected the attributes applicable to the whole model are shown as shown below in Figure 17
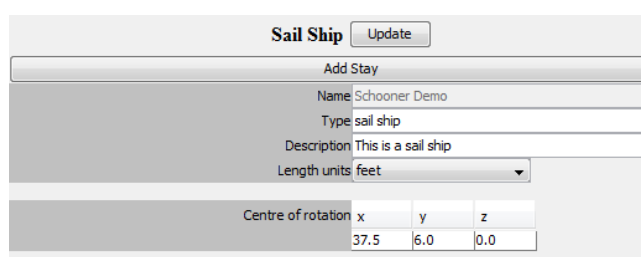


**Figure 17: The top level attribute panel**

Note that one of the attributes is *Length units*. This can be set to be either *feet* or *metres*. **ScenarioBuilder** will scale models appropriately.

## Hiding components to reduce clutter

When building a complex model, it sometimes helps to hide all the parts you are not interested in. Also, a complex model can become noticeably slower to refresh while editing if the graphics card is beginning to struggle.

To address these issues, you can **Right Click** on nodes in the tree view to hide the node (and the components underneath it in the tree.  below shows the result of right clicking on the bowsprit and rudder nodes. The hidden nodes are represented by in the tree view.
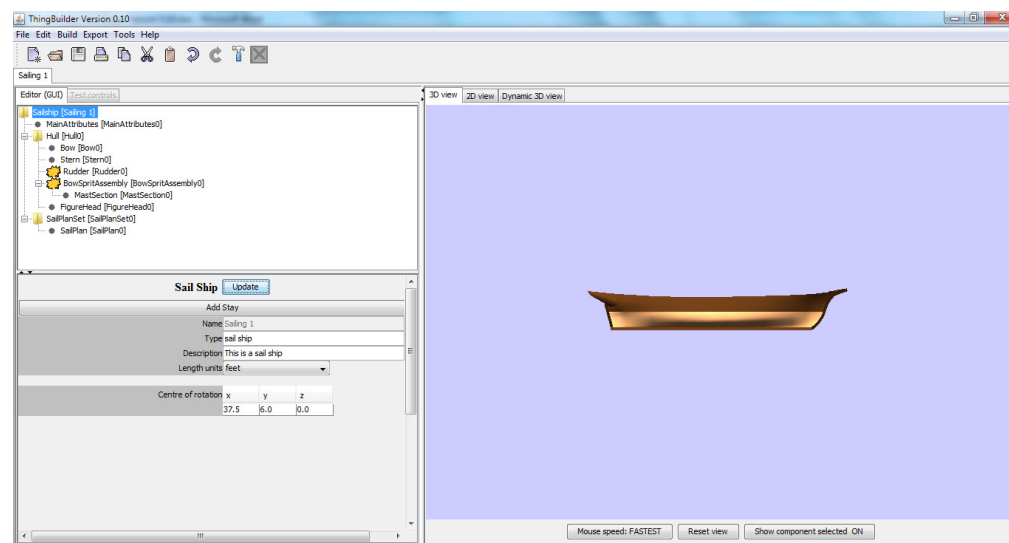


**Figure 18: Hiding the bowsprit and rudder**

You can reverse the process by right clicking again.


The hidden nodes won't actually disappear until you click on the Update button in the selected component's attribute panel. This means that you can quickly right click a whole bunch of nodes without waiting for the 3D view to refresh each time.

The fact that a node is hidden does not prevent it from being saved when saving the model.

## *Customising a component of the new model*

We can customise components by selecting them in the tree view to make the component's attribute panel appear. In our example we select the ship's hull by selecting the hull node in the tree view. Here is the screen after clicking on the hull in the tree view.
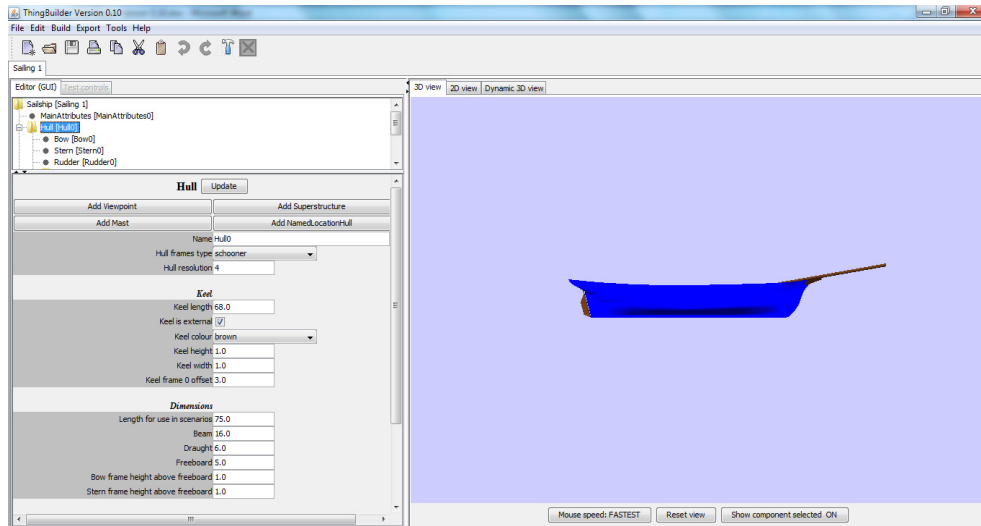


**Figure 19: Selecting the hull**

---

**Note on hiding the selection colour**

Note that the selected component has turned blue. Sometimes this obscures the view - especially when selecting colours. Consequently, a button is provided to toggle switching on and off the blue selection colour. This button is initially labelled **Show component selected ON**.

---

## Modifying an attribute

We can now edit the hull by modifying the attribute values in the panel. The hull length depends on the keel length. We can change its value by clicking in the *keel length* text field and keying in the new value and cpressing the **Enter** key of clicking on the **Update** button. Figure 20 below shows the keel length changed to 100 feet.
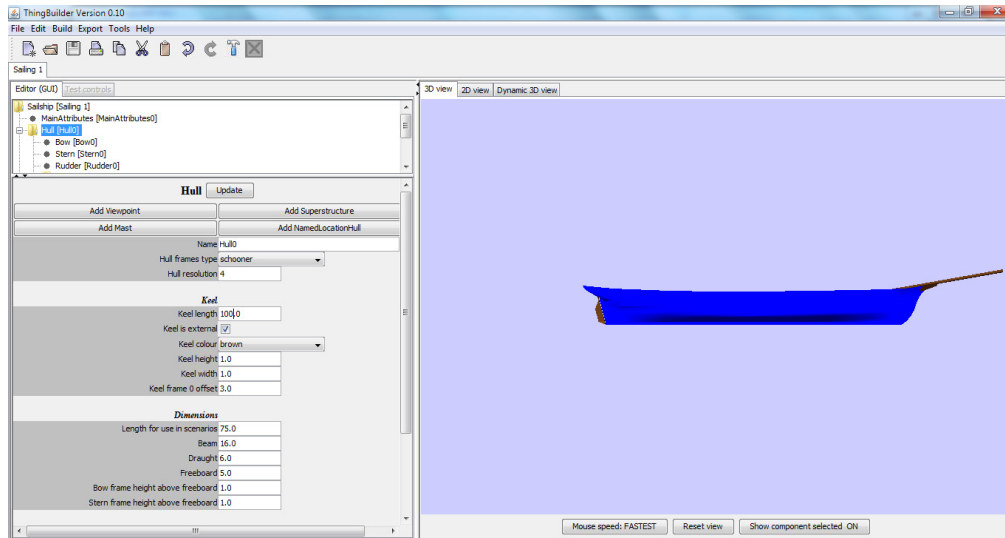


**Figure 20: Hull keel length increased**

## Adding subcomponents

Note that in the attribute panel there are a number of buttons labelled **Add Viewpoint**, **Add Superstructure**, **Add Mast**, **Add NamedLocationHull**.

These represent the subcomponents that can be added to a hull. If we click on **Add Mast** we can the model shown below in Figure 21.
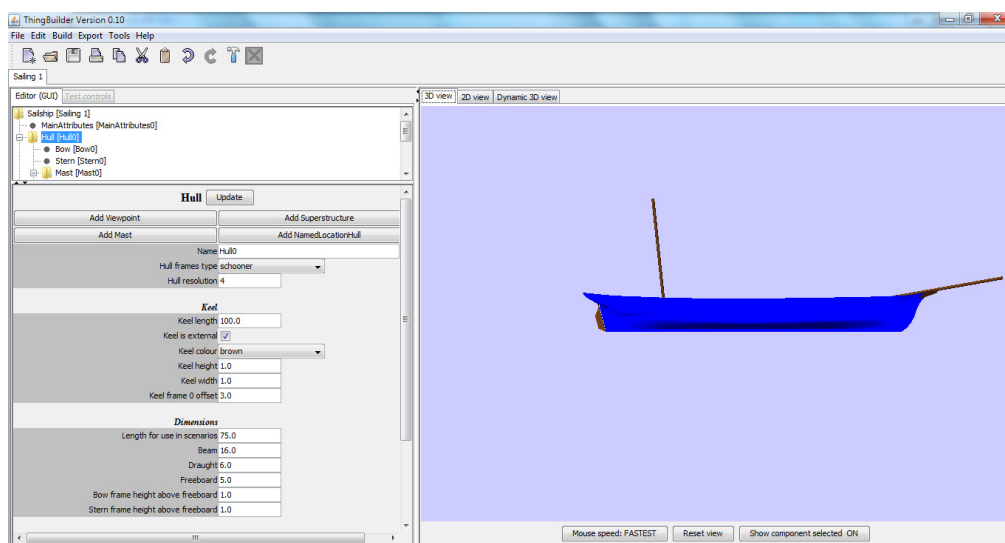


**Figure 21: Adding a mast subcomponent**

Note that the tree now has a mast node under the hull nodes. This node has, in turn, got a mastsection subcomponent.

We can stack mast sections to form a mast and adjust the mast sections to provide overlaps. Thus we can build mast structures with sections overlapping ("doubling") as shown in Figure 22.
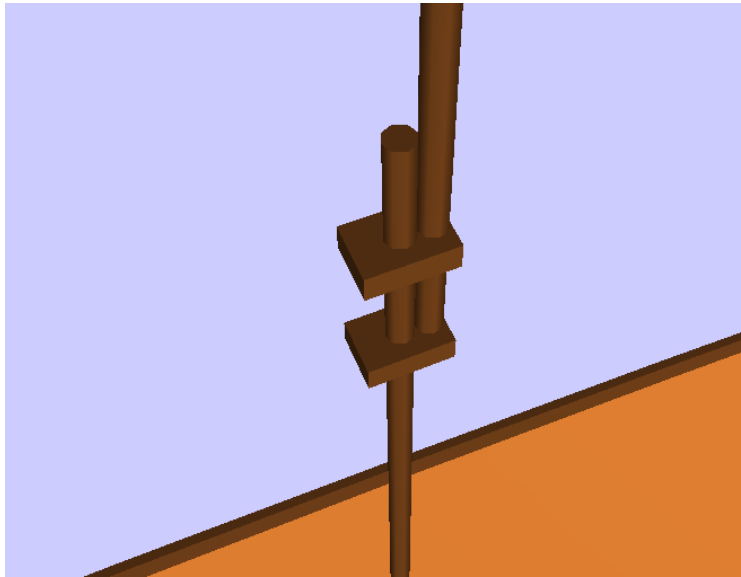


**Figure 22: A mast with two sections doubled**

Figure 22 was achieved by doing the following:

- Select the mast which currently has just one mast section

- Click on **Add MastSection**

- Select the newly added MastSection

- Change the ***Doubling*** attribute to 10.0

- Change the ***Horizontal offset*** attribute to 1.0

- Click on the attribute panel's **Update** button

- Select the original mast section

- Click on **Add SquarePlatform** twice

- Select the first square platform and change its ***Height*** attribute to place it at the foot of the upper mast section

- Select the second square platform and change its ***Height*** attribute to place it further up the upper mast section

## Copying and pasting subcomponents

We can copy a selected component and paste to other components - as long as it makes sense to do the pasting. Thus we can paste a mast to a hull but not to another mast. ThingBuilder only allows what makes sense. We shall now copy and paste our two sectioned mast.

Here is the original mast with the mast selected.
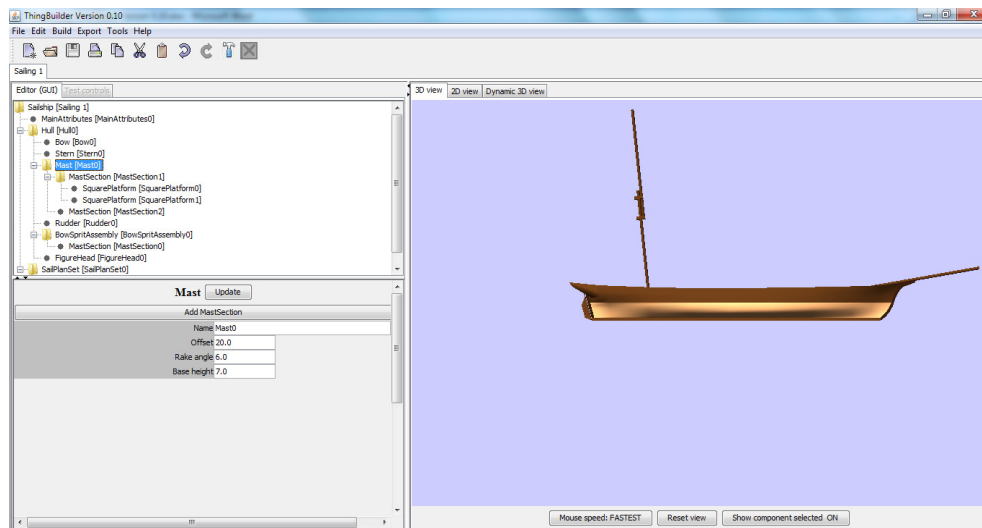


We add four copies by doing the following:

- Select the mast in the tree view

- Click on the copy button  or select menu item **Edit | Copy**.

- Select the hull in the tree view

- Click on the paste button  or select menu item **Edit | Paste** four times

This results in the model looking like Figure 23
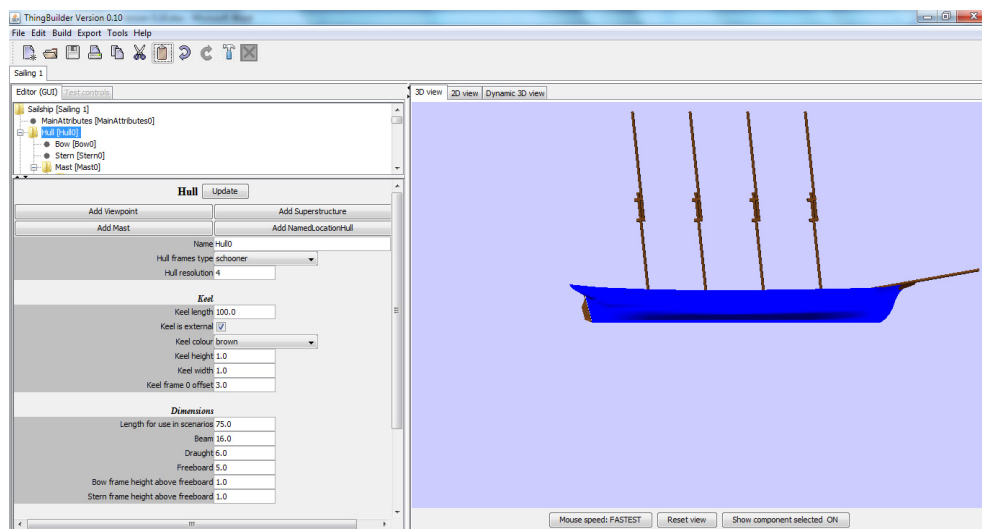


**Figure 23: Four copies pasted**

We can now seelct each mast in turn and customise its position by editing its *Offset* attribute.

If we had tried to paste the mast to a mast we would get a dialog box like Figure 24:
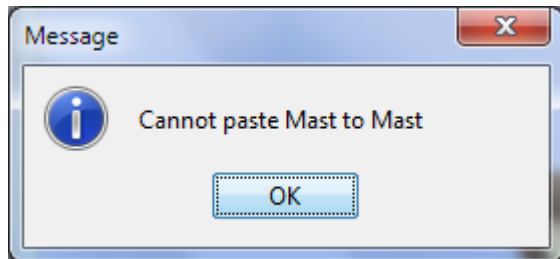


**Figure 24: Illegal attempt at pasting**

## Naming components

When we add a component, **ThingBuilde**r gives it a default name. We can edit this name by selecting it and editing the *Name* attribute. to make it more appropriate.

Note that **ThingBuilder** does not allow two components of the same type to have the same name. If we attempt this then we get a dialog something like   appearing:
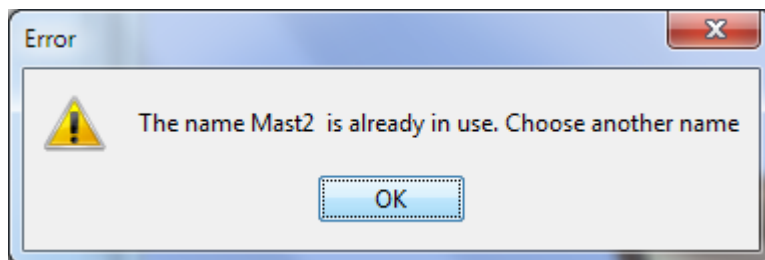


**Figure 25: Name conflict reporting**

## Locations

Certain components, such as stays on masts, need identifiable locations. For ships, two types of named locations are employed:

- *NamedLocationHull*

- *NamedLocationMastSection*

Locations are not visible on the 3D model but are used to identify the ends of **stays** (the ropes that support masts on ships) which <u>are</u> visible.

### *NamedLocationHull*

This location is attached to the hull of the ship. Figure 26 shows a *NamedLocationHull* node in the tree view and its associated attribute panel.
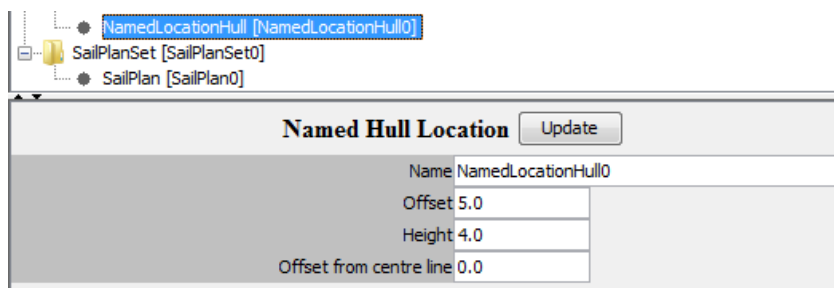


**Figure 26: NamedLocationHull selected**

There are three attributes:

| | |
|---|---|
| *Offset* | Horizontal distance forward from back of keel |
| *Height* | Height above bottom of keel |
| *Offset from centre line* | Horizontal distance to starboard from centre line - negative means it is on the port side |

## NamedLocationMastSection

This location is attached to a mast section. If the mast section is moved then the location moves with it. Figure 27 shows a *NamedLocationMastSection* node in the tree view and its associated attribute panel. Note that the location is a subcomponent of mast section *MastSection10*.
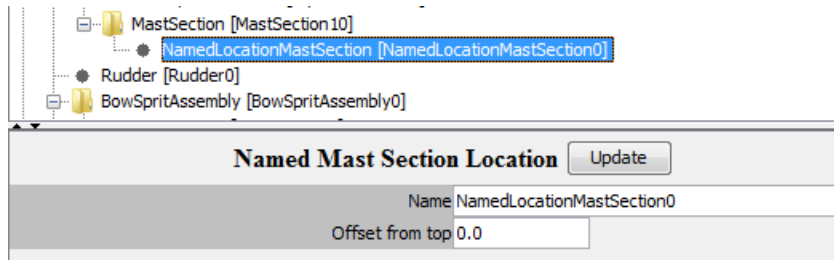


**Figure 27: NamedLocationMastSection selected**

There is only one attribute:

| *Offset from top* | Distance from top of mast section |
| --- | --- |

## Locations and Stays

Locations can be used to specify the ends of stays. Supose we have added to *NamedLocationMastSections* to the top of the foremast and to the bow sprit respectively. Suppose their names are *LocationForeTop* and *LocationBowSpritTop.*

We can add a stay connecting the two by doing the following:

- Select the top level node in the tree view - the ship itself

- Click on the **Add Stay** button in the SailShip attribute panel.

- Move down the tree view to click on this new stay's node. The attribute panel appears as shown below in Figure 28
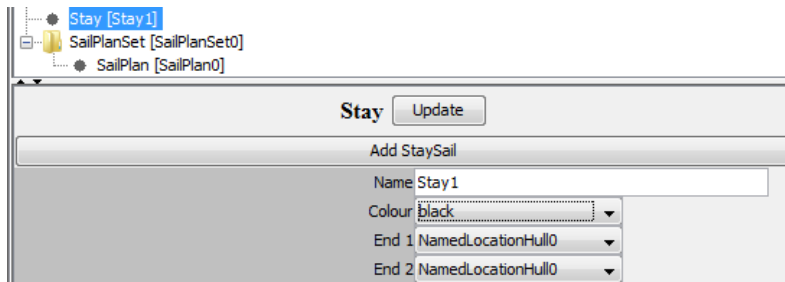


**Figure 28: New stay's attribute panel**

- Click on the End1 and End2 combo boxes and select the locations *LocationForeTop* and *LocationBowSpritTop* respectively. The attribute panel will now look like  Figure 29 below.
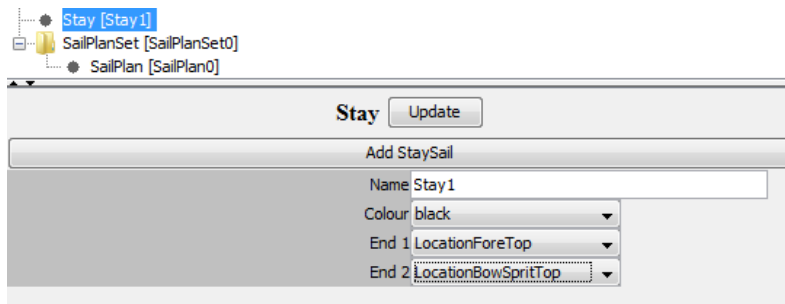


**Figure 29: New stay's ends specified**

The stay is shown in Figure 30  below.
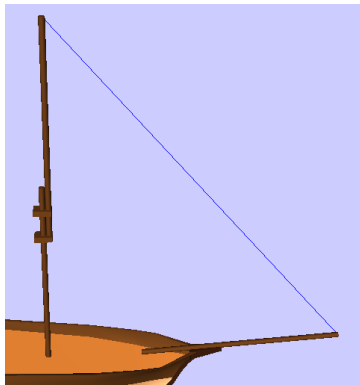


**Figure 30: The new stay in place**

## Sailing ship sail plans

Sailing ship sails are furled an unfurled according to the state of the ship. There are several possible ship states:

| Ship state | Sail status |
|---|---|
| Berthed at a jetty | Sails are generally furled |
| Anchored | Sails are generally furled |
| Assisted berthing | Sails are generally furled |
| Assisted sailing from a berth | Sails are generally furled |
| Sailing | Sails are set furled or unfurled according to wind speed and direction and ship speed |

**ScenarioBuilder** determines the state of the sails of a sailing ship according to the ship's current state. Each type of state has one or more associated **sail plans**. A **sail plan** specifies for each sail the percentage unfurled it is. Thus, in port, anchored, under tow during an assisted docking or sailing the sails may be 0% unfurled while they may be 100% unfurled while sailing.

> Much needs to be done with making realistic **sail plans** e.g. different types of weather and sea state, tacking and wearing will require different plans.
>
> At present the sails are fully set while sailing and furled while berthed. Anchoring etc has yet to be handled.

ThingBuilder gives a sailing ship one sail plan when it is created. With no masts or sails on the ship we get the sail plan shown below in Figure 31.
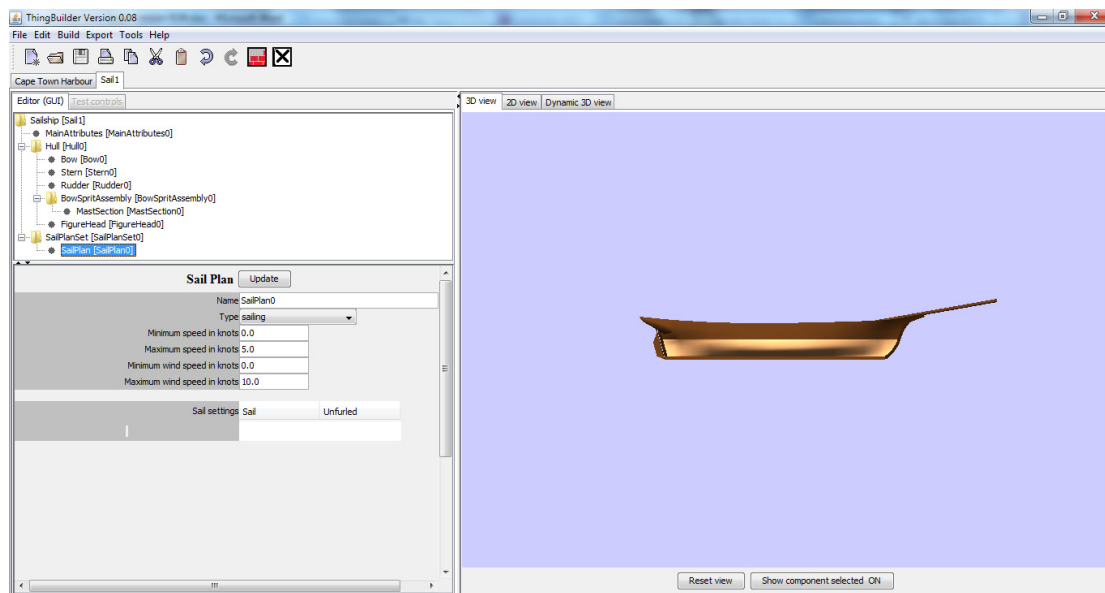


**Figure 31: Sail plan for ship with no sails**

When sails are added, the sail plan setting list will be automatically updated with the sail names and the default 90% unfurled value.

There are four types of plan:

- sailing

- berthed

- towed

- anchored

The default sail plan is for sailing. If the model is imported into **ScenarioBuilder** without any further plans being added, **ScenarioBuilder** will give it a default plan for each non-sailing type needed. **This is still under development.**

Figure 32 shows a sail plan for a small schooner.
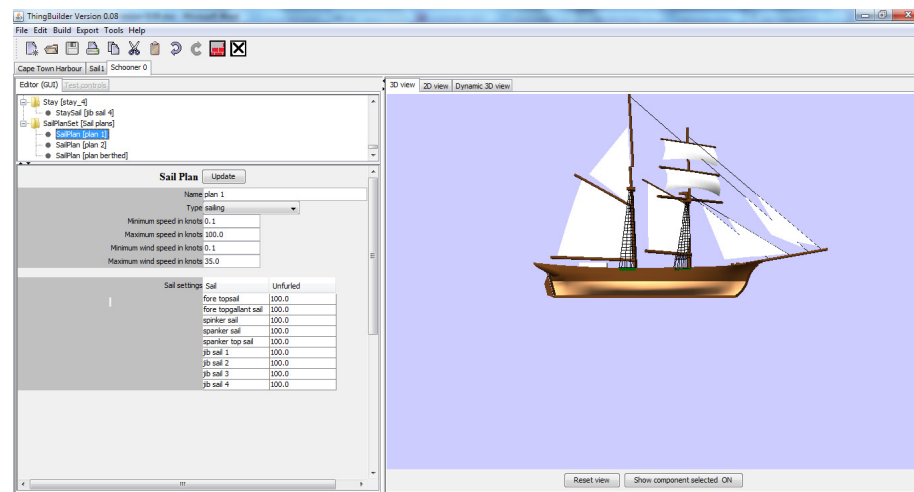


**Figure 32:One of a schooner's sail plans**

There are several attributes:

| *Name* | Each sail plan needs a unique name |
|---|---|
| *Type* | There are four types:<br><br>&bull; sailing<br><br>&bull; berthed<br><br>&bull; towed<br><br>&bull; anchored |
| *Minimum speed in knots* | Used to select this sail plan |
| *Maximum speed in knots* | Used to select this sail plan |
| *Minimum wind speed in knots* | Used to select this sail plan |
| *Maximum wind speed in knots* | Used to select this sail plan |
| *Sail settings* | For each sail in the ship the name and the percentage unfurled.<br><br>The percentages can be edited. |

**Use of these is still under development.**

# Making and testing the dynamic model

So far the 3D model has been written as a single VRML file in the subfolder *model*
The file has no moving parts. We shall now make a dynamic model in which the
rudder and a gaff sail attached to the aft mast can be made to move as in real life.

We add the dynamic gaff sail to the aft mast by selecting the botton mast section then
clicking on **Add GaffRig**. The result is shown below in Figure 33.
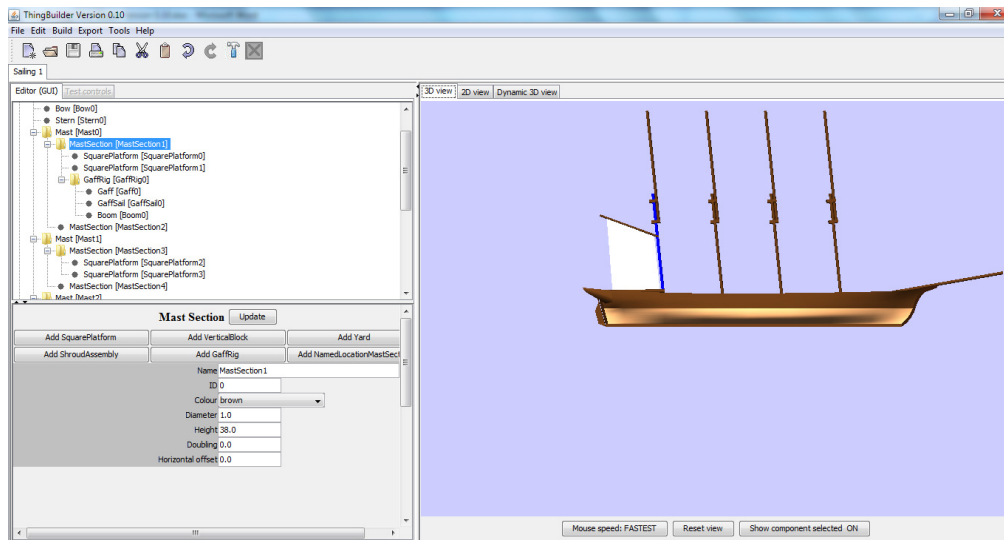


**Figure 33: Adding a dynamic component**

Make the dynamic model by clicking on the button  or selecting the menu item
**Build | Build Dynamic Model**. This results in the screen shown in Figure 34:
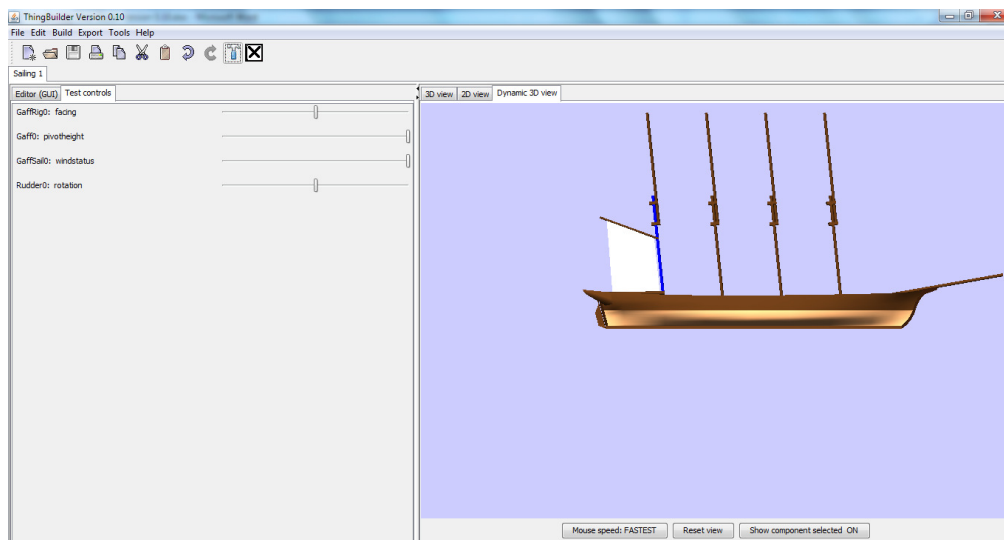


**Figure 34: Building the dynamic model**

The left hand panel, shown in more detail in Figure 35, contains sliders which control the position of the dynamic components.
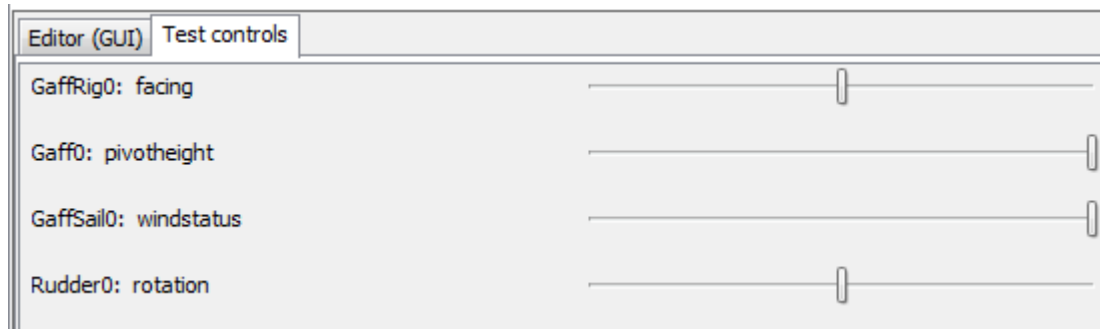


**Figure 35: Dynamic component controls**

Figure 36 below shows the effect of moving the gaff rig facing slider

.
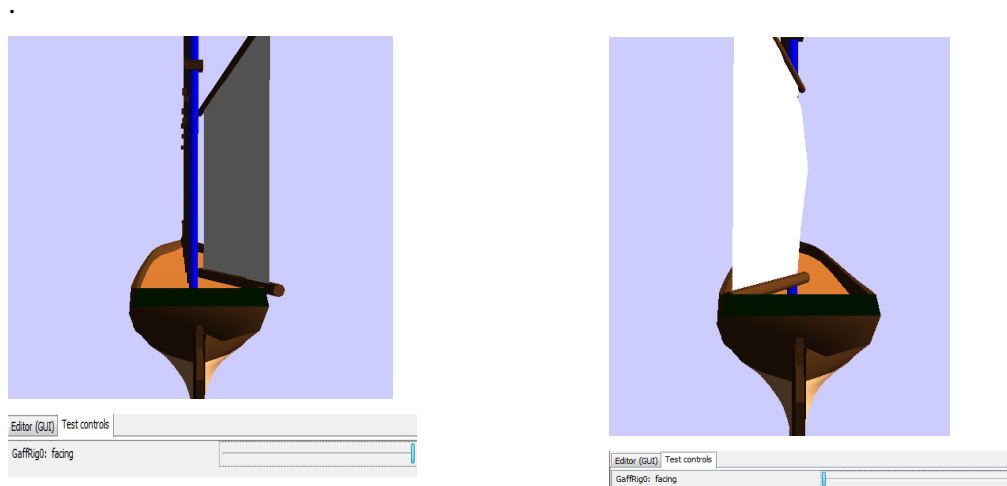


**Figure 36: Moving the dynamic component**

# Exporting a dynamic model

Once the dynamic model has been created it can be exported to ScenarioBuilder for use in scenarios. This is done either by clicking onthe button ☒ or selecting menu item **Export | Export Model to ScenarioBuilder**.

# Scenario Scenes

Scenarios take place in Scenario Scenes which consist of terrain, sea and installations such as ports with jetties, warehouses and other buildings and swing bridges. A scene is added to a scenario by including in the ScenarioBuilder narrative sentences such as "*Scene is Liverpool*". If no scene is mentioned in a narrative then a default terrain is generated consisting of just sea.

ScenarioScenes are made in ThingBuilder just like other types of model but they have a number of special features which are described in this section.

## The components of a Scenario Scene

Figure 37 on the next page shows the **class association diagram** for Scenario Scenes.

A class association digram is an object oriented software development tool. Each box in the diagram represents a component type and the lines represent associations between components.

- The label 0..* between *ScenarioScene* and *Harbour* indicates that a *ScenarioScene* can have any number of *Harbours*, including none.

- The label 1..* between *Tower* and *TowerSection* indicates that a *Tower* any number of *TowerSections* but must have at least one.

- The label 1 between *ScenarioScene* and *Terrain* indicates that the *ScenarioScene* must have a *Terrain* - and only one of them

- The label 0..1 between *Terrain* and *Sea* indicates that a *Terrain* can have no *Sea* or one *Sea*.

- The label +2 between *Lock* and *LockGate* indicates that a *Lock* must have exactly two *LockGate*s.
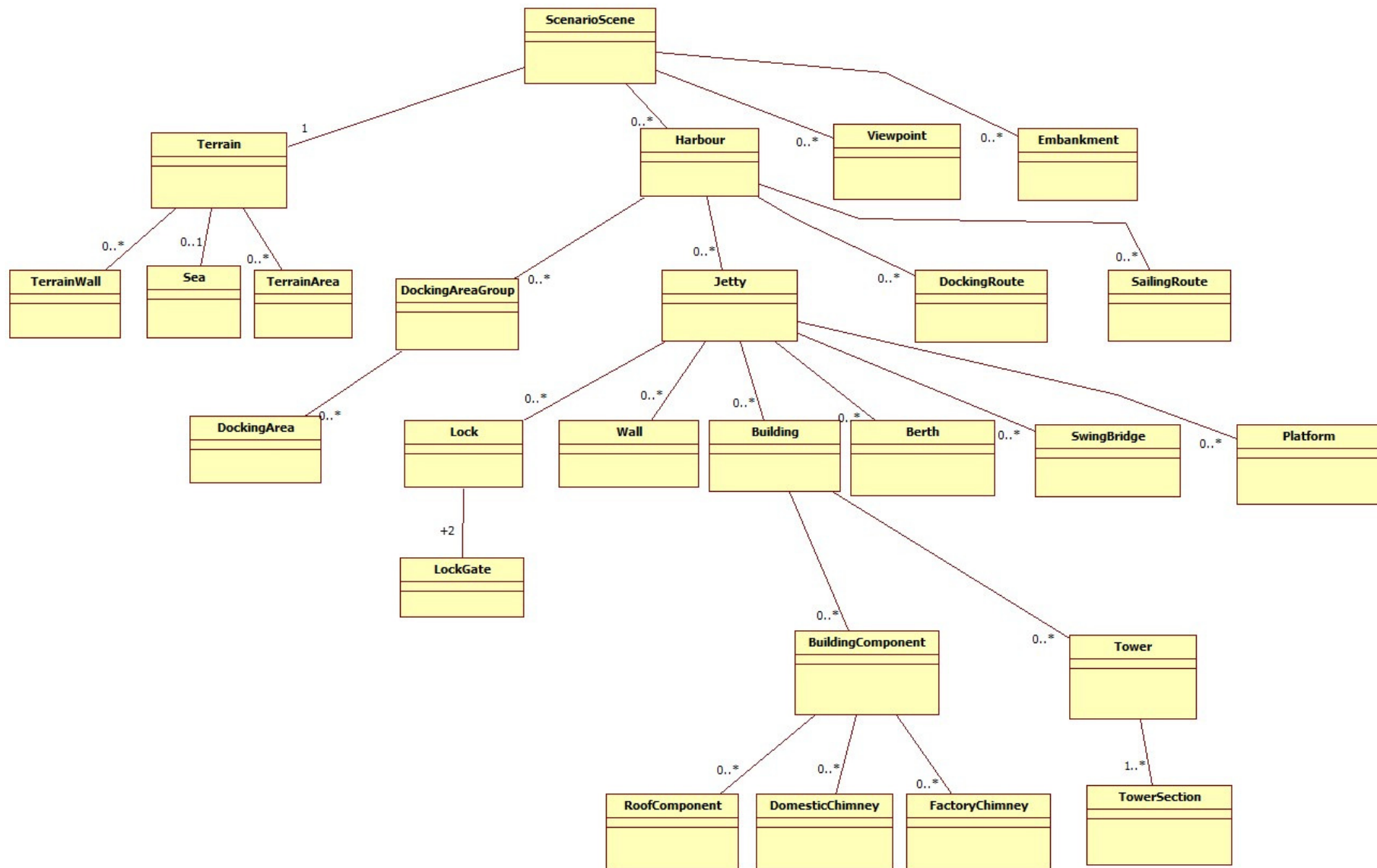
**Figure 37: Class assocation diagram for ScenarioScene**

## *Making a new scene*

When a new scene is created, ThingBuilder looks like Figure 38:
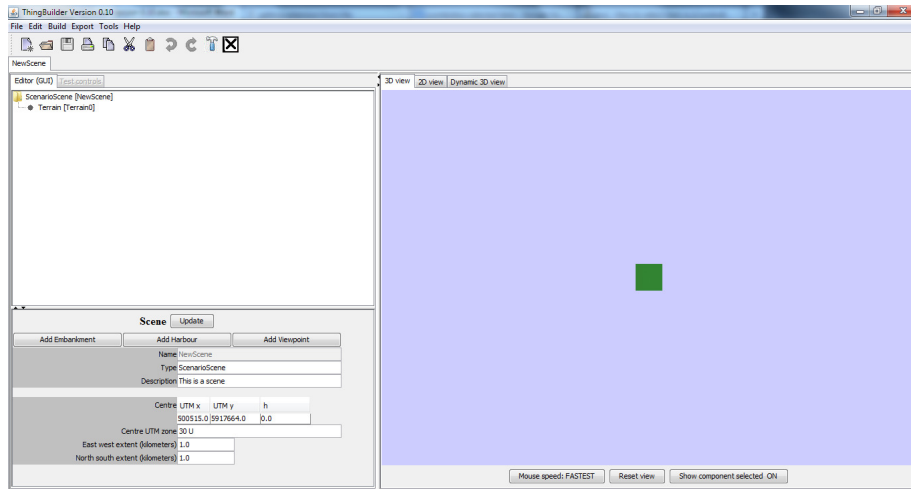


**Figure 38: New scene in ThingBuilder**

## *Terrain and sea*

## Terrain

A terrain is automatically added to a ScenarioScene. [This fulfills the requirement expressed in the class diagram that a ScenarioScene must have one Terrain component.]

The 3D panel is a bird's eye view of a 1 kilometre square of terrain. The terrain is centred at the default scene centre of UTM coordinates *30 U 500515.0 m E, 5917664.0 m N* which happens to be in the centre of Liverpool.

The bottom left panel has the terrain attributes. If we change the *East west extent* and *North south extent* to 10.0 km each then click on the **Update** button we get the bigger terrain shown in Figure 39. Generating the new terrain can take some time for reasons that will be explained below.
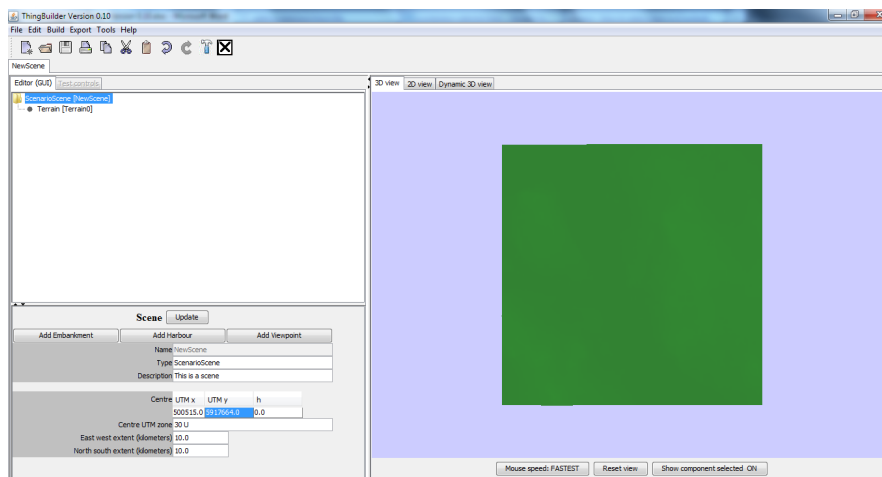


**Figure 39: Increasing the terrain extent**

## *Viewing the terrain in 2D*

In the 2D view the terrain is displayed as a grid of coloured rectangular points as shown below in Figure 40.

- each rectangle represents a point in the GIS data.

- points above sea level are coloured green; points below sea level are coloured blue.

- 
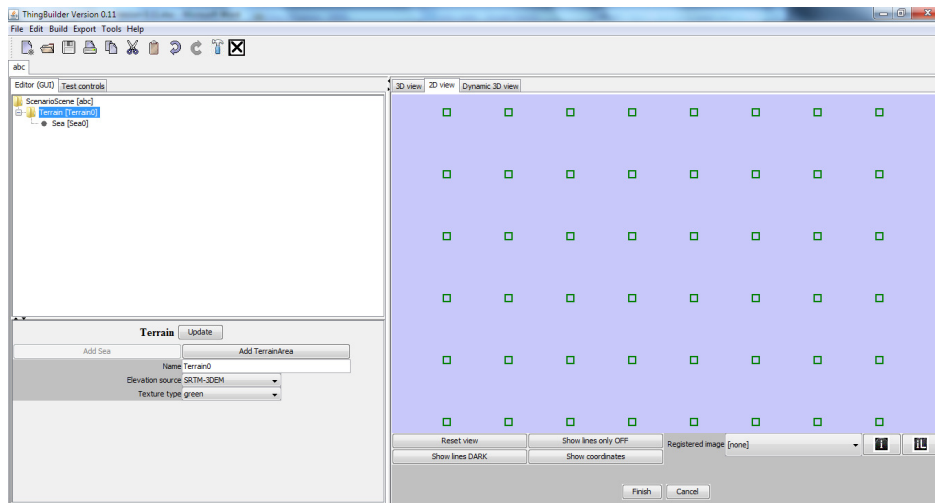


**Figure 40: Terrain grid shown in 2D**

You can zoom out and use the right button to move the terrain in the 2D view.

## Adding the sea

In the model tree, select the terrain node. This will open the terrain attributes panel with the button **Add sea** available.

Click on this button and you will be shown the 2D view of the terrain and invited to add points for the sea. Do this by clicking on a sequence of points in the 2D view. Click on the **Finish** button to complete the sequence. The simplest sea to draw is a quadrilateral covering the four corners of the terrain. The 3D result is shown below in Figure 41
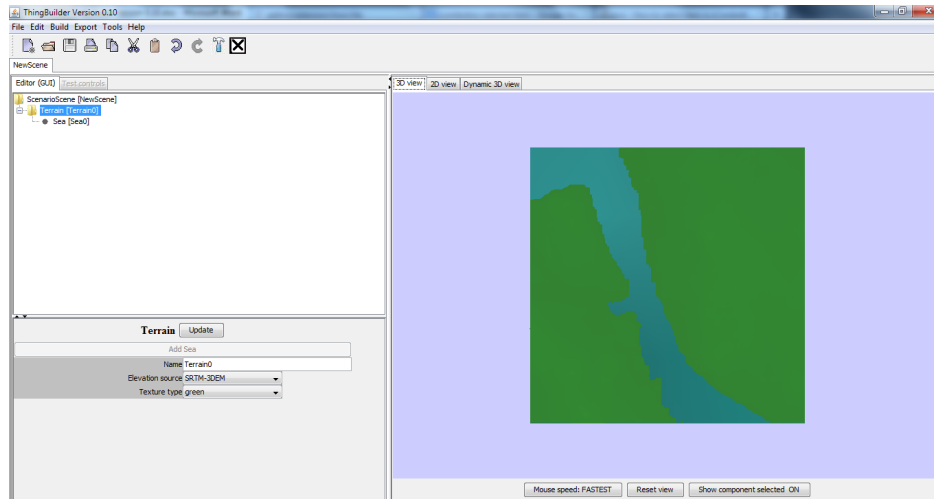


**Figure 41: Sea added to the scene**

We can now see the outline of the River Mersey estuary.

Note that the **Add sea** button is now disabled. This meets the requirement that we cannot add more than one sea to a terrain.

The *Sea* includes the following attributes:

- sea type
- tide height (meters)

If you change the sea type you will change the sea colour

If you increase tide height, the sea will rise to cover more land. Decreasing the tide height reveals more land.

## Sources of terrain data

The terrain data is currently obtained from the **Shuttle Radar Topography Mission** (SRTM) (http://www2.jpl.nasa.gov/srtm/). The data was obtained by the space shuttle *Endeavour* during an 11-day mission in February 2000. The resolution was 1 arc second (30 metres) but this version has only been released for the United States terrain. The rest of the world is released at 3 arc seconds resolution (90 metres). The data can be downloaded and pre-processed so it can be used in ThingBuilder.

For the installation of ThingBuilder, you are provided with some of the data (north of England).

## Copyright for SRTM data

The copyright situation for SRTM data is described at
http://www2.jpl.nasa.gov/srtm/mou.html

## Missing terrain data report

If you change the Scene's *Centre* attributes to a region which is not in the provided data, you will see a message box appear as shown below in Figure 42.
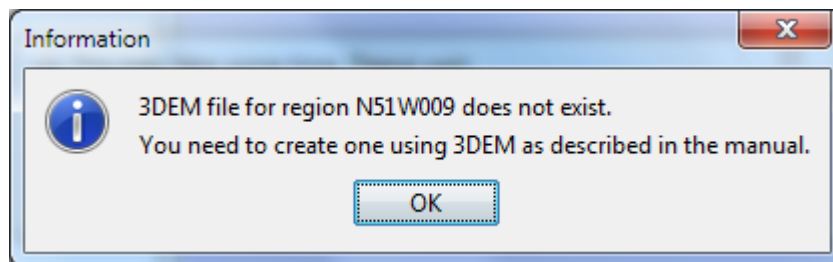


**Figure 42: Missing terrain data report**

If you want to model terrain that is not in the provided data you can get it and preprocess it from the SRTM web sites as described in the next section.

## Obtaining SRTM data

SRTM data is divided up into files each containing a1 degree square of terrain. The file names denote the latitude and longitude of the degree square's bottom left corner. Thus, the missing area reported in the message box in Figure 42 has name *N51W009* which means that the file contains the data from latitudes and longitudes bottom left 51 N, 9 W to top right 52 N, 8 W. Thus we need to download and pre-process the corresponding file from the SRTM web site:

(http://dds.cr.usgs.gov/srtm/version2_1/SRTM3/)

The web page (as of Feb 2013) is shown below in Figure 43
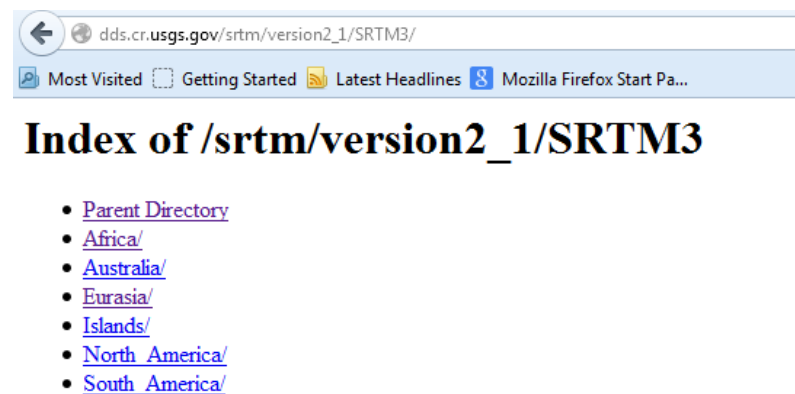


**Figure 43: Web page for index to SRTM data**

Our area of interest is Eurasia so we click on that link to the page shown below in Figure 44.



**Figure 44: Web page for Eurasia SRTM data**

We find *N51W009.hgt.zip* on this page. Download it.

Copy the downloaded file to the SRTM_data folder inside the ThingBuilder installation folder (typically c:\ThingBuilder\SRTM_data if you installed ThingBuilder in the default folder).

Right click on the file's icon in Windows Explorer and select the menu item **Extract All...** Accept the default destination.

## Using 3DEM to preprocess the SRTM terrain data

We need to get the data into UTM format for use in ThingBuilder. This is achieved using 3DEM software which is freely available on the web at http://www.hangsim.com/3dem/

Download and install 3DEM and launch it.On launching 3DEM you will see the dialog shown below in Figure 45.
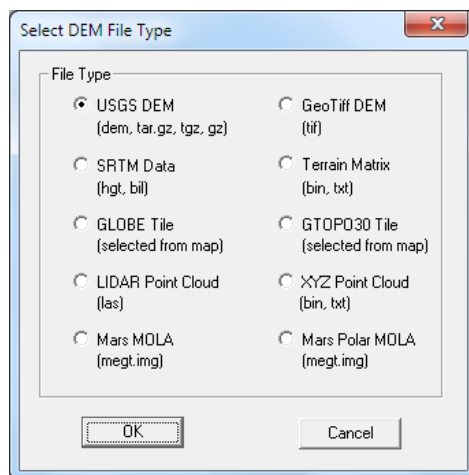


**Figure 45: Selecting DEM File type**

Select **SRTM Data (hgt, bil)**

You will then be prompted to open a file of this type with a **File Open** dialog. Navigate to the folder where you extracted the .hgt file as shown below in Figure 46.
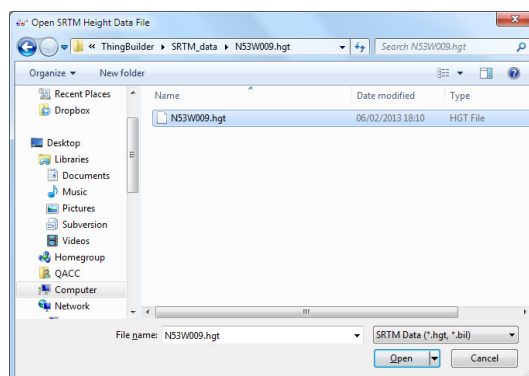


**Figure 46: Opening the .hgt file in 3DEM**

You will see the data mapped as shown below in Figure 47.
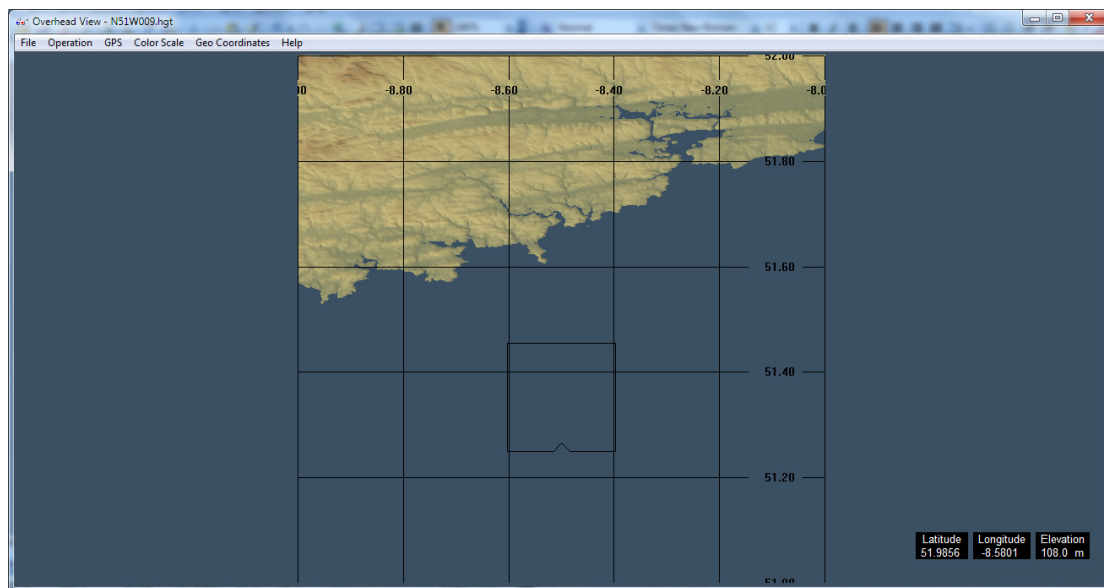


**Figure 47: Southern Ireland terrain in 3DEM**

This data is in degrees af latitude and longitude. We need it in UTM formt.

Select the menu **item Operation | Change Projection | Convert to UTM Projection**.

Accept the default *WGS84 UTM Ellipsoid*. The screen changes to that shown below in Figure 48.
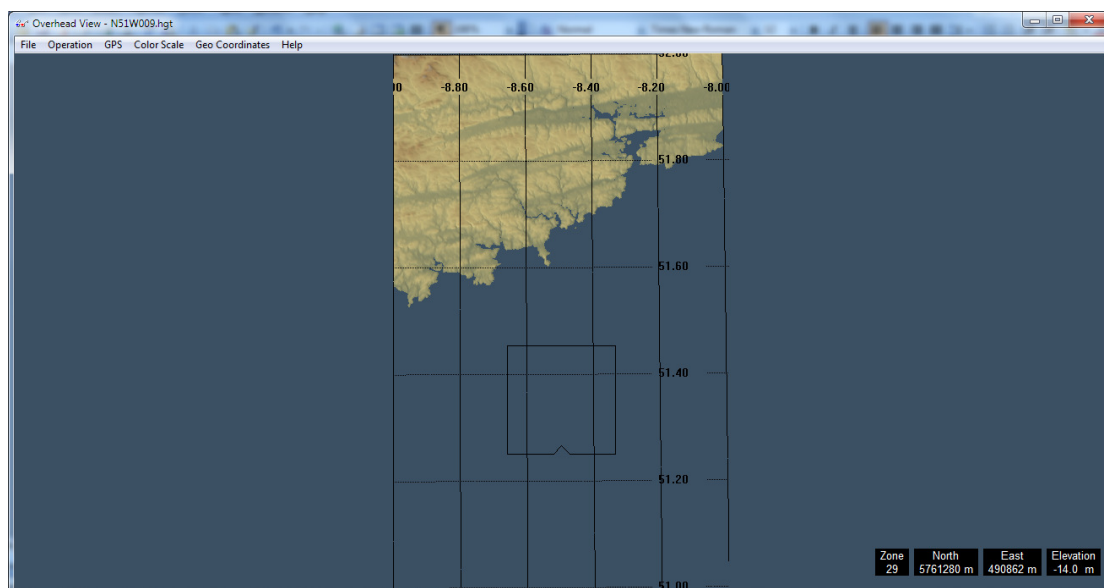


**Figure 48: 3DEM terrain in UTM projection**

Save the terrain by selecting the menu item **File | Save Terrain Matrix** selecting *ASCII Text String* for the **File Format**. Use the text name *N51W009.txt* so that ThingBuilder can find it in the folder *ThingBuilder\SRTM_data\N51W009.hgt* .

If you now retry the **Update** button in ThingBuilder you should see the green square.

Add the sea and change the extent both north-south and east-west to 10 km and you should see the last port of call of the Titanic before heading into the Atlantic as shown below in Figure 49.
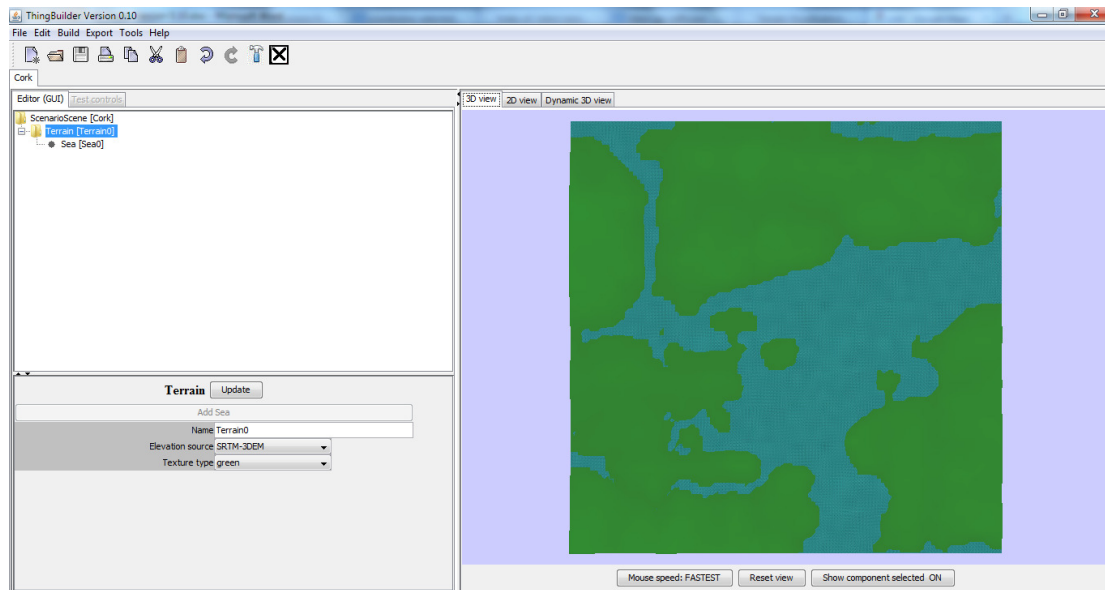


**Figure 49: Lough Mahon near Cork, Ireland**

It is possible that your model includes areas of more than one degree square. ThingBuilder will extract the required data from the several files automatically, asking for missing ones as required.

ThingBuilder converts a fresh degree square into a zipped file for subsequent reading. The generation of a terrain with a new centre or extent can consume a minute or two if the terrain is several kilometers in extent. However, once created, the terrain will be stored in a compact, rapidly accessible format until the centre or extents change again.

## Modifying the terrain with TerrainAreas

SRTM height accuracy can vary according to the type of terrain. Over hilly and built up areas it is less accurate. In order to correct the terrain used in a scene we have added *TerrainArea* objects. The terrain points inside a TerrainArea have their height modified.

A *Terrain* can be associated with any number of *TerrainArea* objects.

Add a *TerrainArea* to the *Terrain* by doing the following:

- Select the *Terrain* in the tree view

- Click on the **Add TerrainArea** button

- In the 2D view click on the sequence of points determining the polygon covering the extent of the area.

- Click on the **Finish** button when done.

The polygon can be modified by selecting the *TerrainArea* in the tree view and dragging, adding and deleting points.

The TerrainArea has one attribute:

- Height difference over terrain

This attribute value is added to the height of each terrain grid point enclosed by the TerrainArea polygon.

Judicious use of TerrainAreas can be used to "improve" the accuracy of terrain.

## 2D modelling with Scenario Scene building

Geographical models such as scenario scenes have some of their components edited using the **ThingBuilder** 2D view. This includes structures such as jetties and buildings.

We shall use the ScenarioScene started on page 32 which is repeated below in Figure 50.
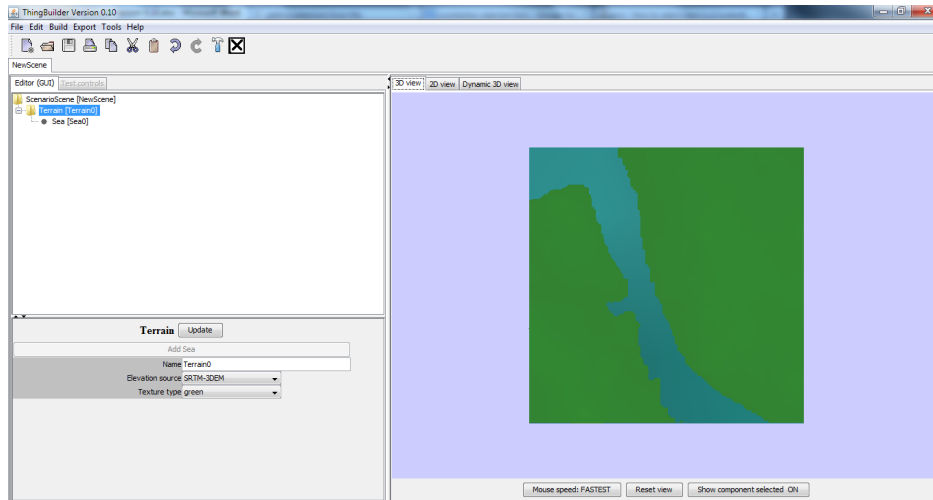


**Figure 50: Editing a scenario scene**

## *Adding components in the 2D view - building a harbour*

### Adding a harbour

To add a harbour to the scene, we select the top level node and then click on the Add Harbour button in the attribute panel.

A dialog box pops up as shown below in Figure 51.



**Figure 51: Adding a new harbour**

When you click on OK, you are taken to the 2D view tab as shown below in
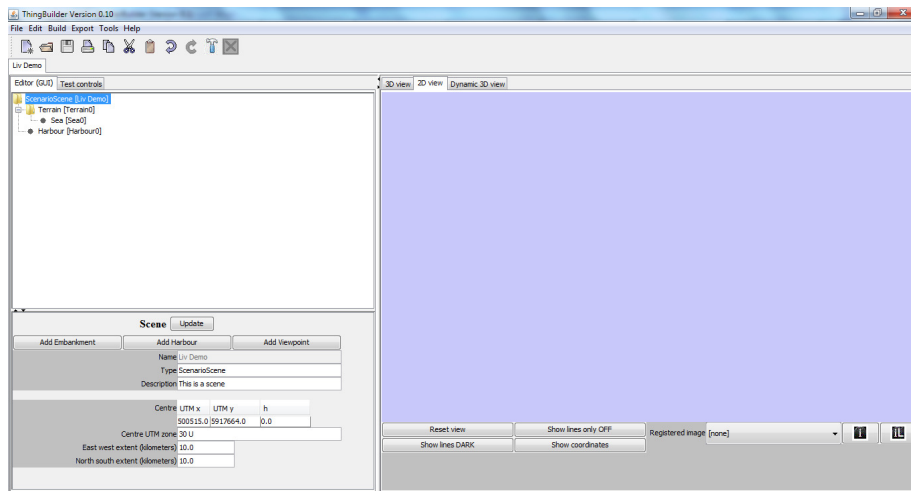
**Figure 52: The 2D view**

Clearly we need some guidance as to where to place the harbour. We shall make use of a registered image provided with the installation of ThingBuilder.

## Setting up a backdrop image

At the bottom right of the 2D view there are the controls shown in more detail below in Figure 53.



**Figure 53: Registered image controls**

Registered images are added to the ThingBuilder library as described on page 12. The images in the library are listed here.

this button refreshes the list of the all the images in the library

this button refreshes the list only with those images which overlap the extent of the scene - it cuts down the clutter of irrelevant images.

If you use the **Tools** menu to add an extra image or edit an existing one to help with the model you are working on you can make it appear in the list by clicking on either of the two refresh image list buttons.

We select the River Mersey image for the list. This results in Figure 54.



**Figure 54: River Mersey Google Earth image selected**

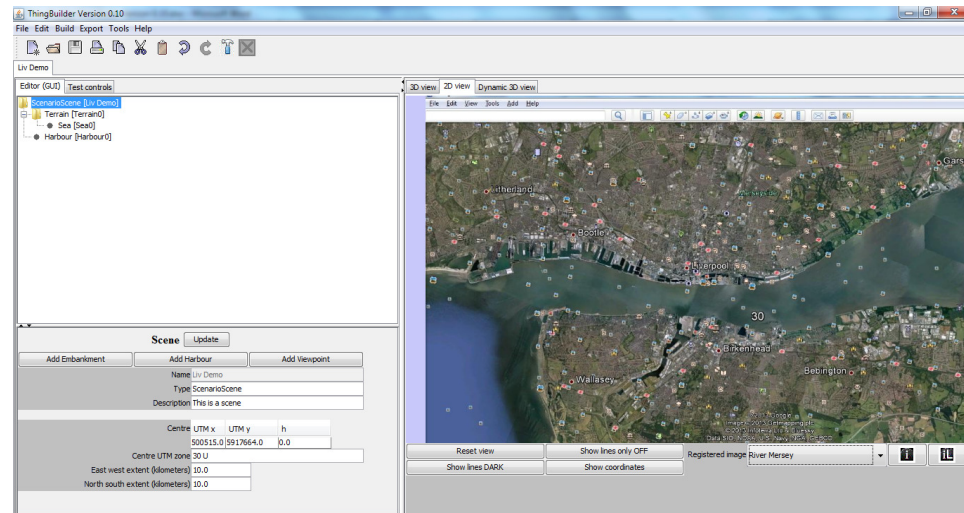Note that the north is not at the top of the image. The view was rotated to get a good stretch of river into the image in a landscape view. ThingBuilder takes care of adjusting for such rotations automatically through the registered image system.

## Capturing images to register

The Google Earth image was made from using **Print Screen** (Alt-PrtSc keys) to put it into the *Copy* buffer on a PC and then pasting it into MicroSoft Paint. The file is then saved. It was then registered using the menu item **Tools | Register An Image...** as described on page 12.

## Copyright for Google Earth images

The copyright situation regarding use of these images is given at http://support.google.com/earth/bin/answer.py?hl=en&answer=21422 .

Old maps have also been scanned in and registered to provide images for other models.

## Moving around the 2D view

You use the mouse to move around the 2D view:

| | |
|---|---|
| **Left button click** | Select points and lines on components |
| **Right button down drag** | To move the model in the view |
| **Centre wheel button down drag** | To zoom in and out |
| **Centre wheel** | To zoom in and out |

## Selecting the harbour location

When we click on the 2D view, to position the harbour you will see the following dialog box in Figure 55:
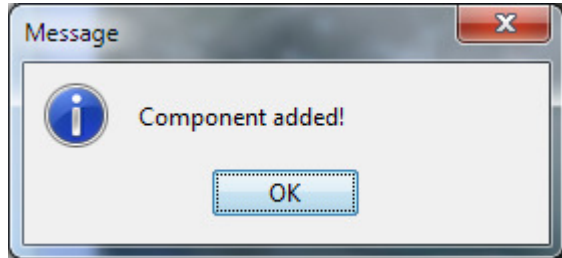


**Figure 55: Confirmation of a component being added to the 2D view**

There isn't actually anything to see of the harbour in the 2D view but it does appear in the tree view. This is because the harbour will be made up of visible subcomponents as they are added. There is nothing to show for it at this early stage.

The selected point will provide the centre of rotation when the harbour node is selected in the tree view.

We are now ready to add a visible component to the harbour.

## *Adding a jetty to the harbour*

Select the harbour node in the tree view.

Click on **Add Jetty** button in the attribute panel.

We can now start clicking on the corners of the jetty in the 2D view. In Figure 56 we can see that we are using another Google Earth image showing more detail of the Albert Dock and have clicked on a sequence of points as recorded by the white squares and lines.
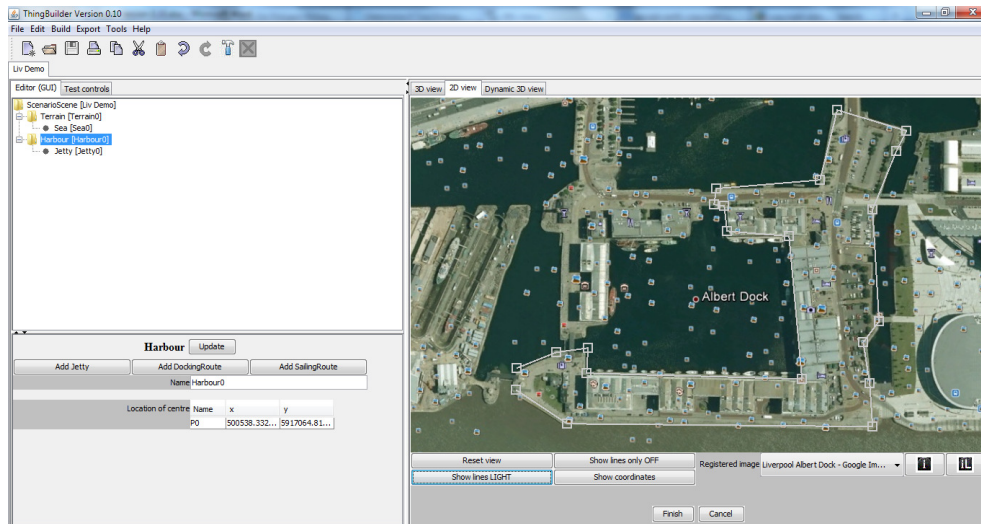


**Figure 56: Adding the corners of a jetty**

If we click on the **Finish** button the jetty will be completed as shown in Figure 57.
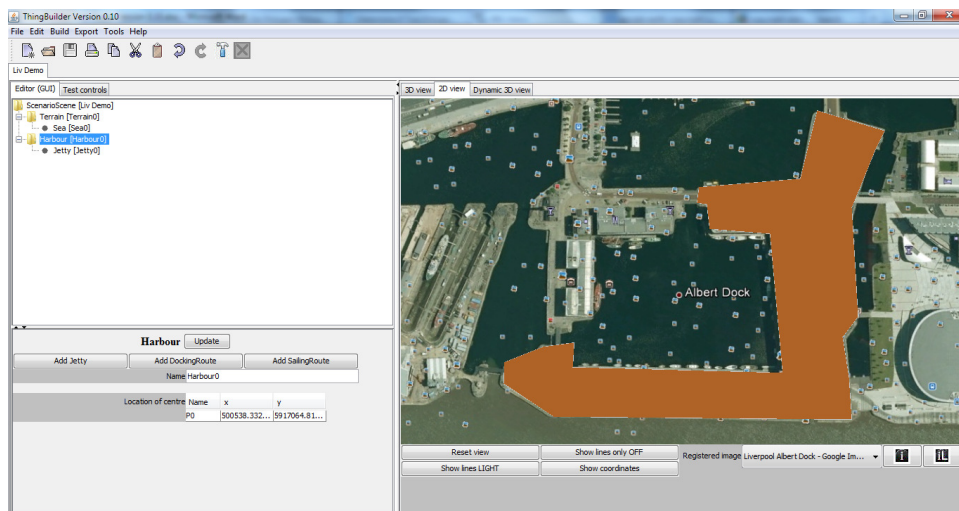


**Figure 57: Completing the jetty**

There are a number of extra controls in the 2D view designed to improve the usability. These are shown below in Figure 58.



**Figure 58: 2D view usability buttons**

| | |
|---|---|
| **Reset view** | Restores the view to the initial view. |
| **Show lines only OFF / Show lines only ON** | Toggles showing jetties etc coloured in. This button helps you to see the image underneath the components. |
| **Show lines DARK / Show lines LIGHT** | Background images may be dark or light. This button toggles showing component lines in dark or light colours to improve visibility. |
| **Show coordinates / Hide coordinates** | It may sometimes be useful to know the UTM coordinates of a point in the 2D view. This button enables/disables their display. |

If we click on the 3D view tab and zoom in we can see our jetty in the terrain as shown in Figure 59.
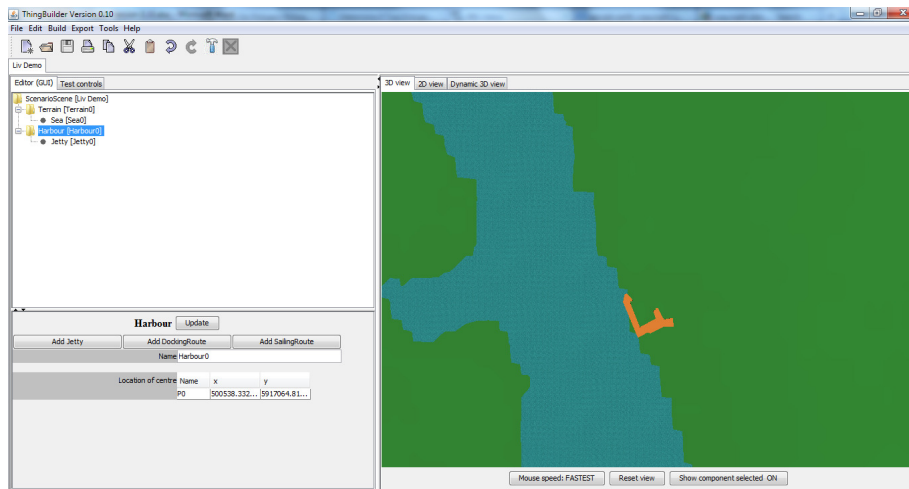


**Figure 59: Jetty in the terrain**

Rotating the view gives us Figure 60.
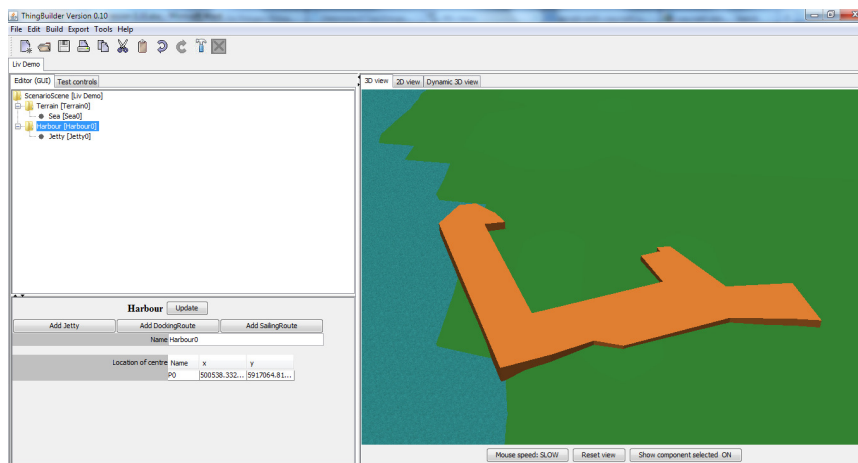


**Figure 60: Close-up of the jetty**

Note that the tide is out! We can control the height of the sea dynamically as we shall see in the next section.

## Dynamic model of the scene

Click on menu item **Build | Build Dynamic Model** or the toolbar button [image] to build the dynamic model. It will have one slider control that sets the sea height. Raisning the sea height results in Figure 61.
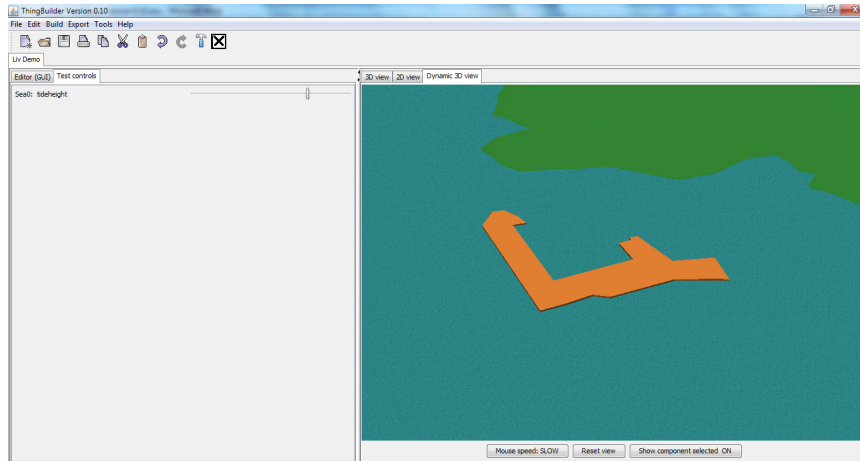


**Figure 61: High tide at Albert Dock**

The jetty is surrounded by water. Clearly more building work needs to be done!

## *Editing the jetty in 2D*

Editing the jetty by moving points, adding points or deleting points is quite straightforward.

You can select the jetty in two ways:

- by clicking on its node in the tree view
- by clicking on one of its lines in the 2D view

With a complex model with lots of jetties, clicking on the 2D view becomes the easiest way of selecting.

Figure 62 shows the jetty selected.
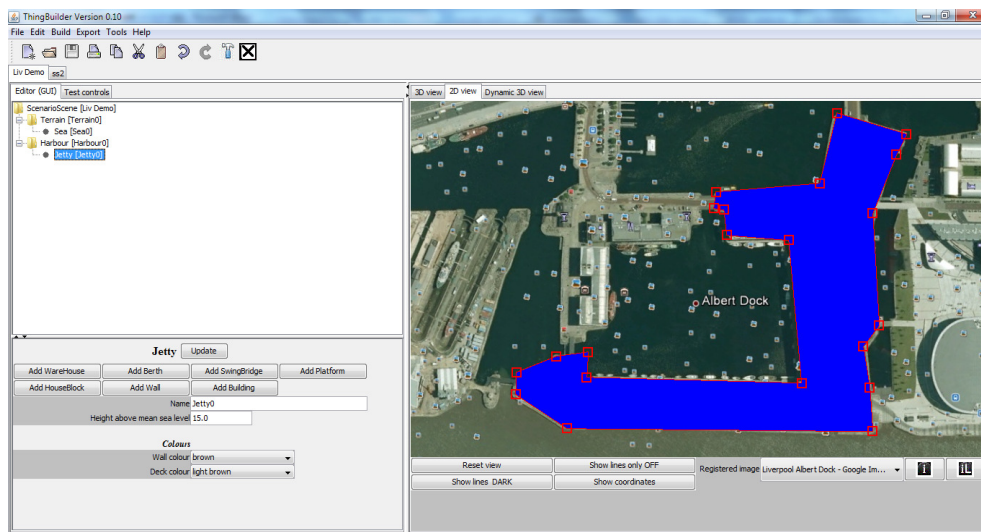


**Figure 62: Jetty selected**

Suppose we want to refine the model at the bottom left corner - shown below in Figure 63.
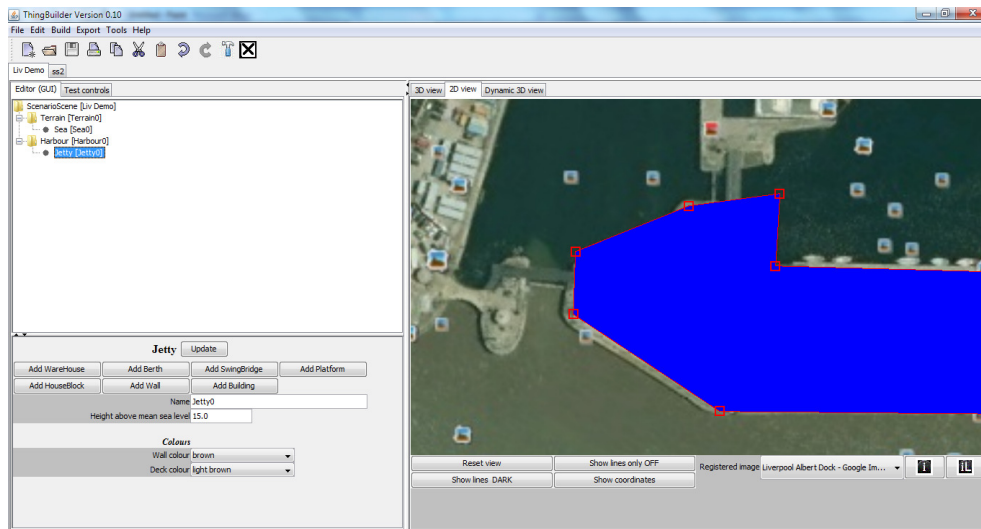


**Figure 63: Part of the selected jetty**

We can see that our initial plotting of jetty points has not been very accurate.

To get a clearer view we click on Show lines only OFF to reveal the image underneath as in .
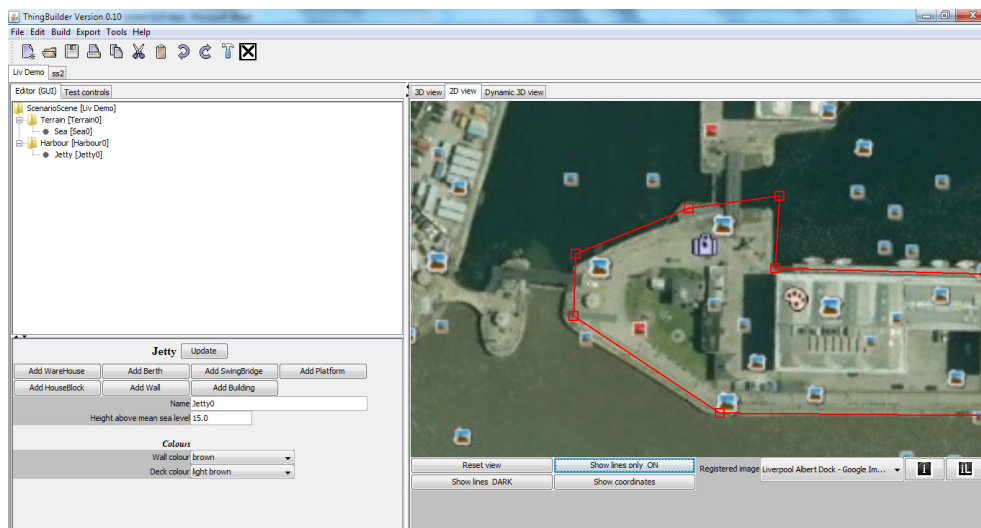


**Figure 64: Revealing the image beneath the jetty**

## Moving points

We can now use the mouse with the left button down to drag the points into better positions as shown in Figure 65.
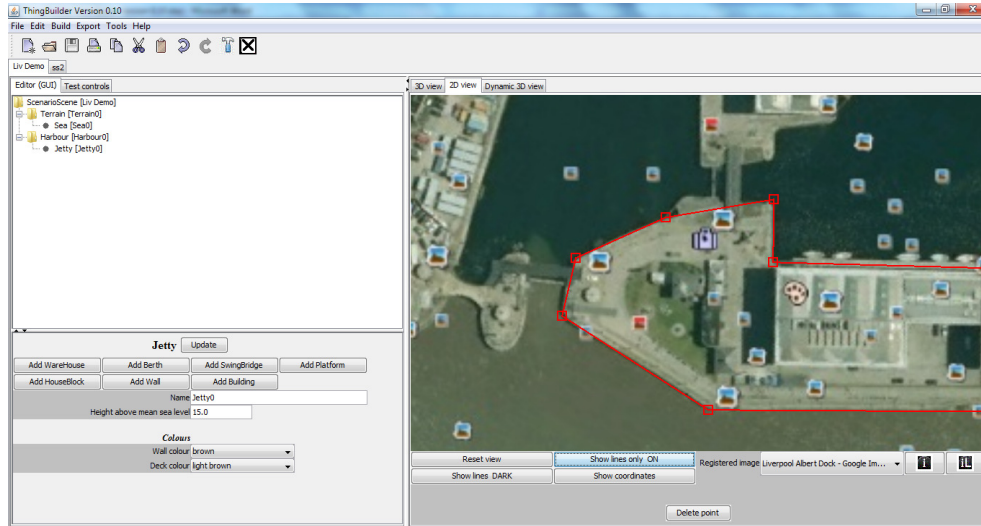


**Figure 65: Improving the jetty points**

## Inserting points

Figure 66 shows a small part of the jetty.
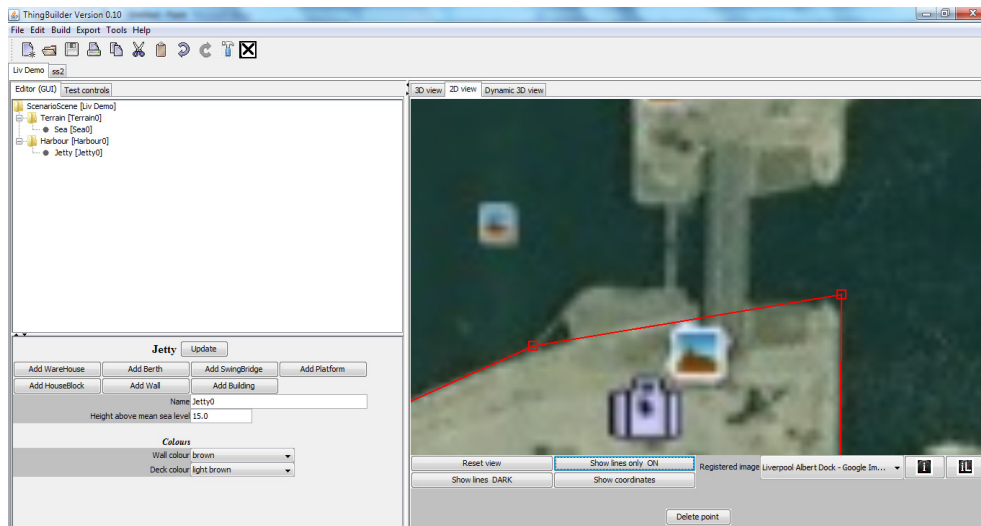


**Figure 66: Jetty detail**

We can see that we have missed a paret of the jetty protruding into the water. We remedy this by doing the following

Select the jetty line by clicking on it. It turns into a thicker line.

Click on **Add point to line** 5 times.

Drag the points to the corners of the protuberance. as shown below in Figure 67 .
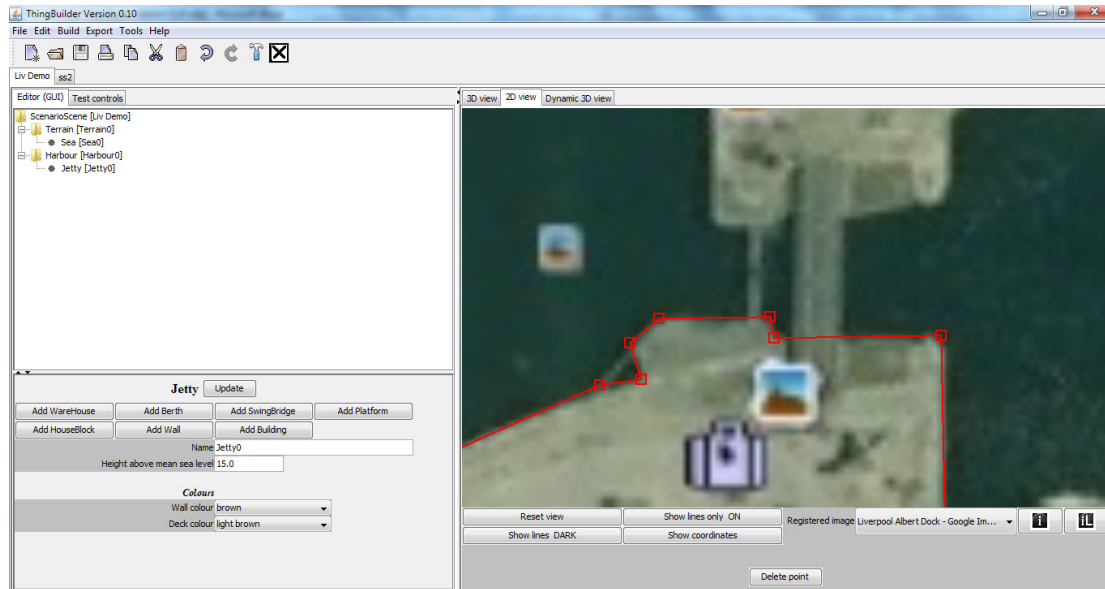


**Figure 67: Adding jetty detail**

## Deleting points

We may have added a point which we do not want. To delete the point, do the following

Select the point with the mouse.

Click on the button **Delete point**.

## Locks and half tide docks

The water level inside a dock will rise and fall with the tide unless locks are provided to keep the water level inside high while that outside falls. An early type of lock still used in harbours is the **half-tide dock**.

The idea of the half tide dock is to keep the gates open while the tide is above half way and closed while the tide is below half way. The effect is to ensure the water level inside the dock never falls below half tide and so the ships stay afloat even though the sea bed outside the dock is uncovered by water. This kind of lock requires just one set of gates as shown in Figure 68.
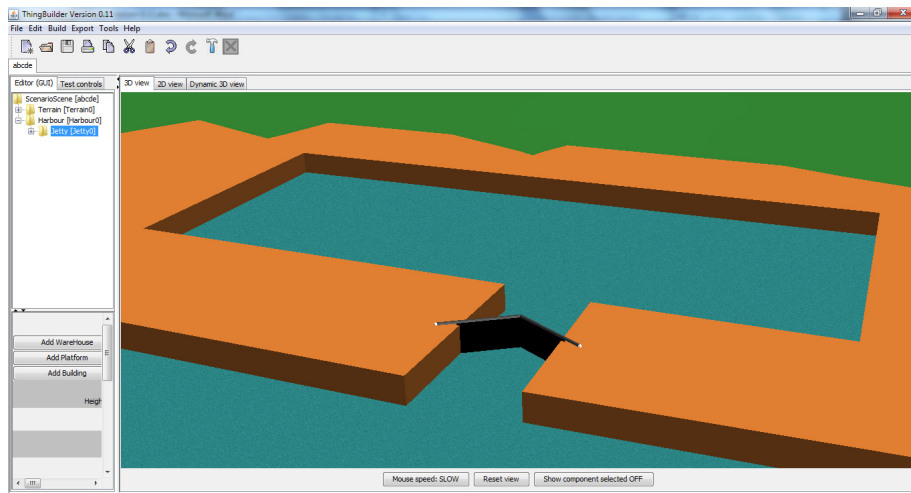


**Figure 68: Half-tide dock with lock**

In ScenarioBuilder, half tide lock gates will be operated automatically as the tide rises and falls.

Different levels of water within and outside of half tide docks require us to add stretches of water to the scene in addition to the sea. These stretches of water are referred to as dock areas and there heights are controlled by Scenario Builder. They are modelled with *DockAreaGroup* and *DockArea* objects.

- A *DockAreaGroup* is a named collection of *DockAreas*.

- A *DockArea*  is a polygon of water that rises and falls under the control of a half-tide lock.

## Making a half-tide dock

A dock can be made half tide by doing the following:

1) Add a lock at the entrance to the dock by doing the following:

- Select the jetty and then clicking on the Add Lock button in ther jetty's attribute window.

- Select the lock and

    o align the lock so it is pointing into the half tide dock.

    o modify the Inter-pivotpoint distance to cover the dock entrance width as shown below in Figure 69
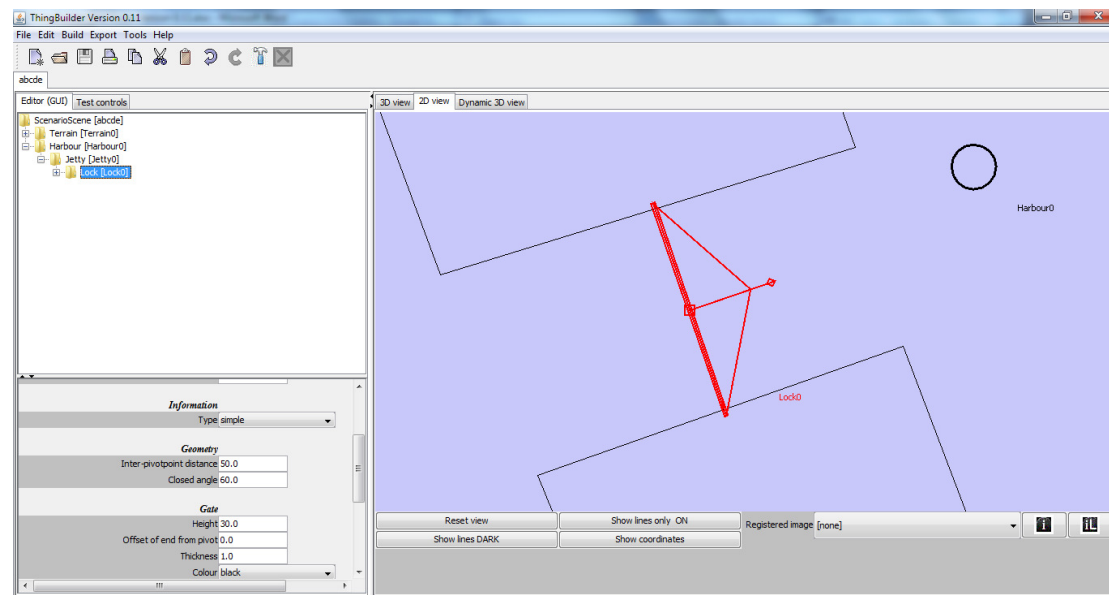


**Figure 69: Modifying a lock**

2) Modify the sea to remain outside the lock gates. This is done by doing the following:

- Select the sea

- Add points to the edge and drag them in to form a boundary at the lock gates as shown below in Figure 70:
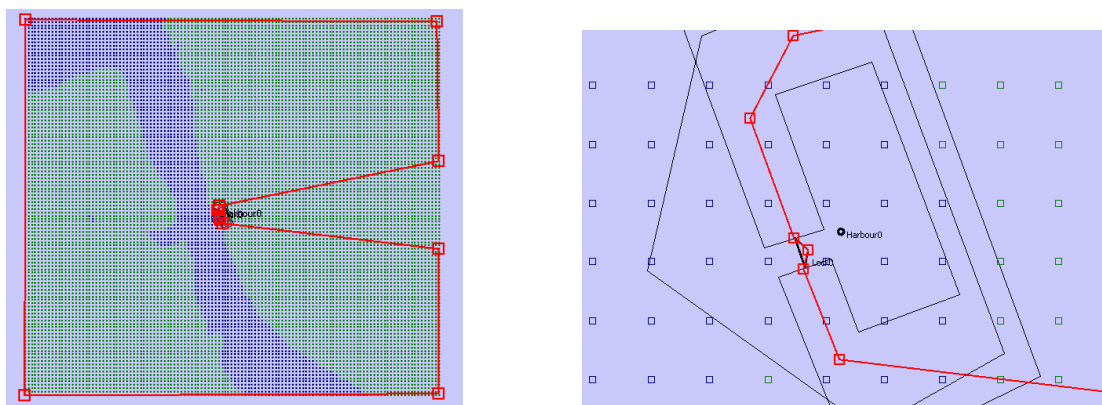


**Figure 70: Modifying the sea boundary - full screen          and zoomed in**

3) Add a dock area for the water inside the dock. This is done by doing the following:

- Add a *DockAreaGroup* to the *Harbour* by selecting the *Harbour* and clicking on the **Add DockingAreaGroup** button in the attribute window.

- Select the *DockAreaGroup* and click on the **Add DockArea** button in the attribute window. This will prompt you to add the points of the dock area polygon. Plot the points on the jetty - there is no need to follow the jetty edge. However, make sure to follow the "pointed" edge of the lock as shown below in Figure 71 where the *DockArea* is selected.

- 



**Figure 71: DockArea points**

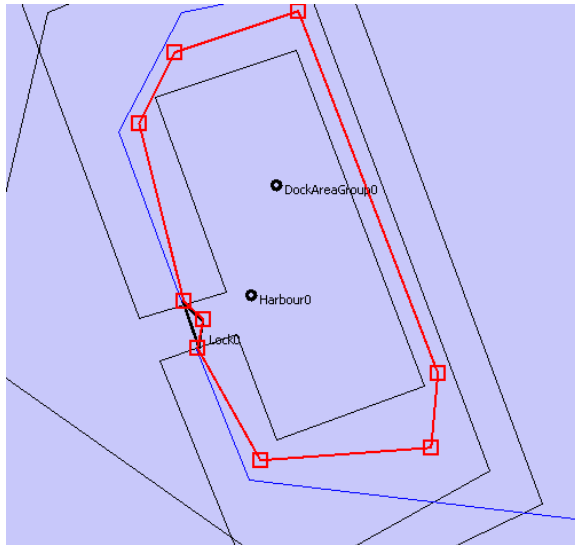The heights of the water inside and outside the dock can now be manipulated separately as shown below in Figure 72.
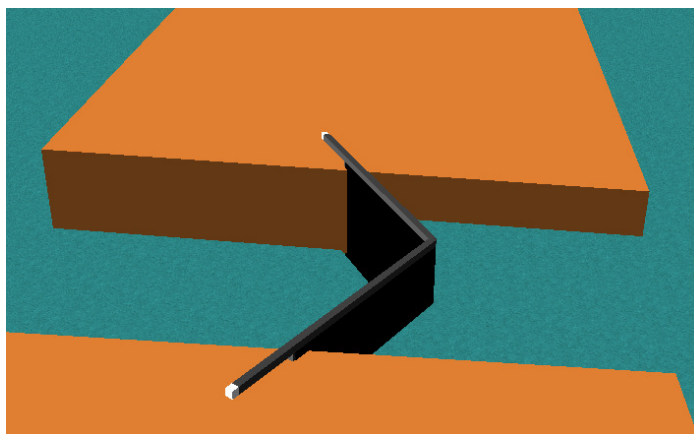


**Figure 72: Water levels inside and outside the dock**

## Adding a Building

Buildings are currently added to jetty components. The Building component, like the harbour component does not have its own geometry. Instead, BuildingComponent components are added. Do the following to add a building.

Select the jetty in the tree view.

Click on the **Add Building** button in the attribute panel. The 2D view appears in which you should click on the centre of the building. This point will be used as the centre of rotation of the building.

Attributes of the building include

- type
- description

These can be used to store information about the building. A future development would be able to read the model file and extract such information.


## Adding a BuildingComponent component

BuildingComponents are added just like jetties in the 2D view. We use the map/Google Earth image to provide the building component's footprint:


Select the Building node in the tree view.

Click on the button **Add BuildingComponent.**


You are now prompted to plot the BuildingComponent points just like you did with the jetty.


## Modifying a BuildingComponent

Selecting the new BuildingComponent reveals that it has attributes including

- side height
- colours
- roof type
- window type

The first two are self explanatory. We shall look at the the last two in more detail.

## Roof type

There are three choices of roof type:

- flat (the default)
- hip
- gable

The last two require attention being paid to the order in which the points of the building component are placed. The following conventions are applied.

a building component has two notional "gable end" walls.

one gable end wall is between the last point plotted and the first point plotted

the other gable end wall is between the two points plotted halfway through the sequence.

For a four point building plotted in the order P0, P1, P2, P3 the 2D lines look like Figure 73 (with the points labelled). The gable ends are the thicker lines.

P3

P2

P0

P1
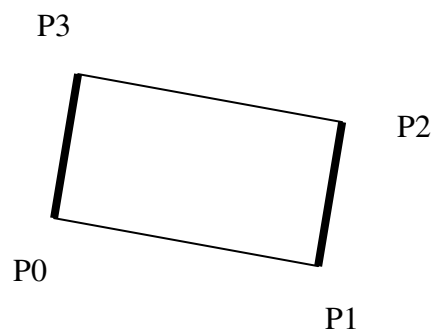
**Figure 73: Gable ends on a 4 point build**

An 8-point building component looks like .

P5

P7

P4

P6

P0

P2

P3
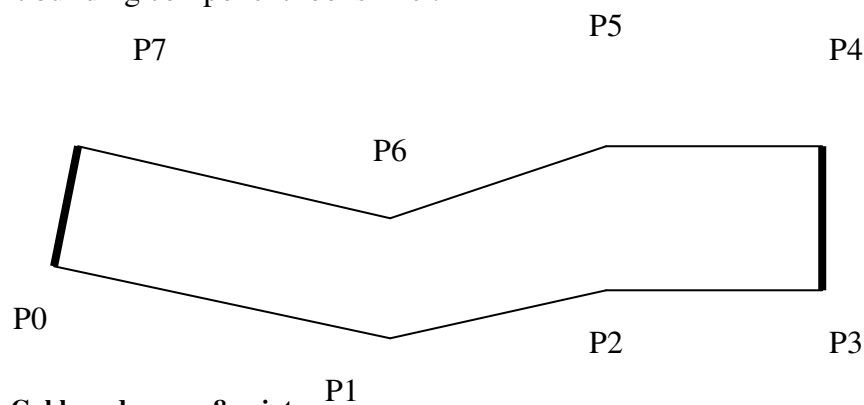
**Figure 74: Gable ends on an 8 point**

P1

.

> Note that gable ends are only guaranteed to produce reasonable looking roofs if the building component has an **even number of points**. You can always add an extra point to make the number even.

> If you get the points in the wrong sequence you can always drag them round to the correct positions.

When you change from a flat roof to a hip or gable roof, ThingBuilder orients the roof to take account of the gable ends as shown in the two exaples shown below in .Figure 75
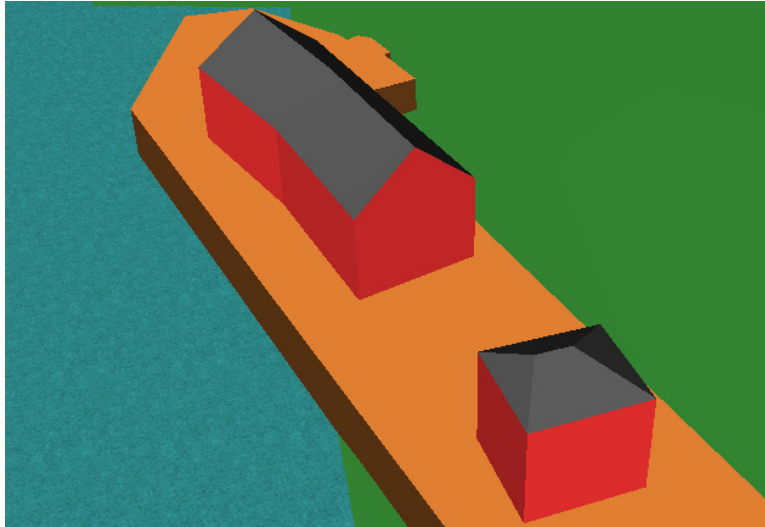


**Figure 75: A gable and a hip roof**

## Roof components

In addition to setting the roof type, we can develop the roof further by adding RoofComponents to the BuildingComponent with the **Add RoofComponment** button.

These are added in the 2D view by plotting the sequence of points.

Once added, they can be customised to form complex roof structures e.g. as shown in Figure 76.
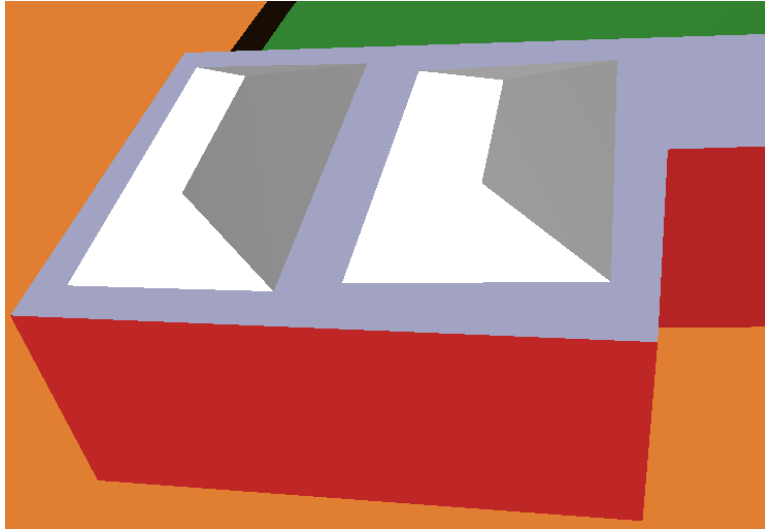


**Figure 76: Roof components on a roof**

## Windows

ThingBuilder enables rapid creation of large numbers of simple windows to speed up building. Currently there are two alternatives:

- no windows
- simple windows

**Simple windows** has these attributes:

- *Floor height*
- *Window width*
- *Window height*
- *Window gap ratio*
- *Colour*

*Window width*, *height* and *colour* are self explanatory.

*Gap ratio* is the fraction of window width that forms the horizontal gap between windows.

*Floor height* is the height of each floor in a building.

ThingBuilder uses these values along with the length and height of the walls to work out the number and positions of each window on each wall.

## Towers

Towers are added to a building by selecting the Building and clicking on the button **Add Tower**.

A Tower is made of a stack of TowerSection components. A new Tower always starts with one TowerSection.
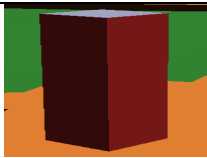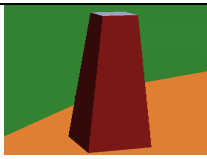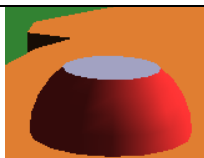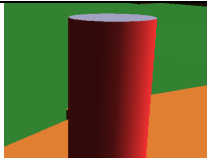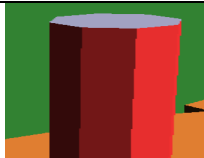
The widths of the sections is specified as follows:

- the Tower has a *Base width* attribute

- each TowerSection has a *Base width indent* attribute. This determines how narrower the section is compared with the section below. In the case of the bottom section it is how much narrower compared with the Tower's *Base width*.

- a negative *Base width indent* means the section overhangs the section below

The idea behind this approach is that we can change the base width of a tower and the sections conform, maintaining the indent but with automatically changed width.

## TowerSections

There are several types of tower section:

| square | | cone | |
|---|---|---|---|
| square frustum | | dome | |
| square pyramid | | dome frustum | |
| round | | octagon | |
| round frustum | | octagon frustum | |

## Windows and openings in tower sections

The following types of tower section can have windows:

- square,

- square frustum,

- round,

- round frustum,

- octagon,

- octagon frustum

The windows have the same attributes as those for Building section simple windows. However, where the base width is different from the top width then the windows are stretched to conform.

Note that openings are implemented by choosing windows of appropriate shading.

## Chimney

Two types of chimney are available for BuildingComponents:

### Domestic chimney

This has a rectangular cross-section with an array of chimney pots on it. Figure 77 shows a house with two domestic chimneys
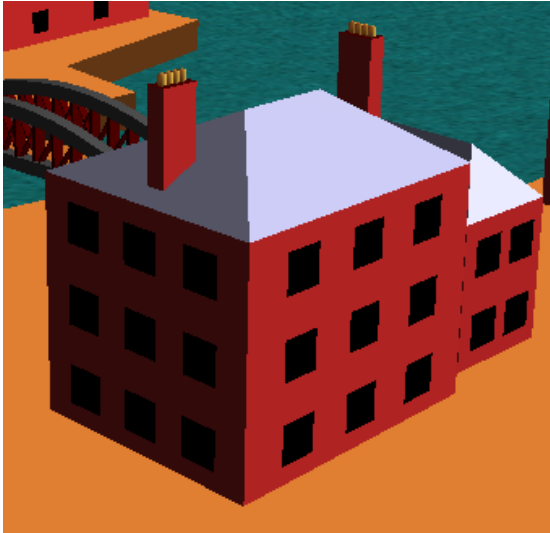


**Figure 77: Domestic chimneys**

### Factory chimney

This is a rectangular or conical cross section as shown in Figure 78.
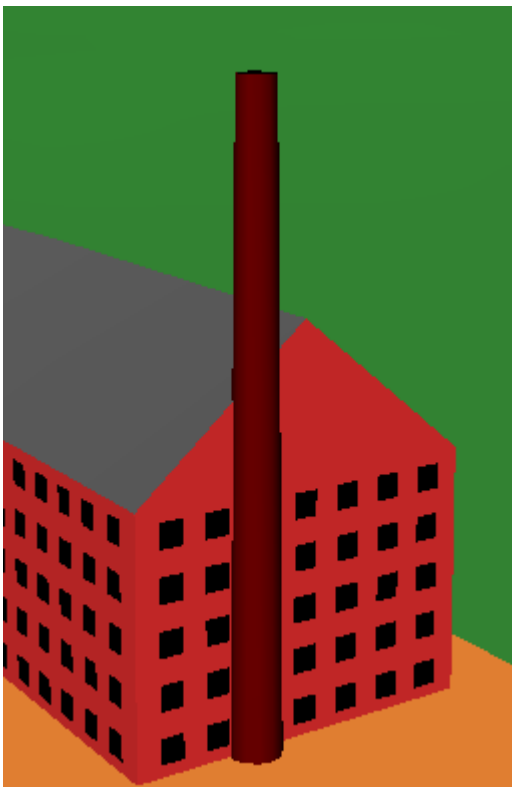


**Figure 78: Factory chimney (conical)**

## *Changing the direction of a component in the 2D view*

Certain components such as chimneys, berths and swingbridges have a *direction* or *heading* attribute that can be modified in the usual way by changing a number in the attribute panel. To aid rapid editing of these attributes, a facility has been added to the 2D view of such components to change the heading visually.

When such a component is selected, the 2D view shows the selected component with a square at the centre of rotation and a handle which can be used to rotate it as shown below in Figure 79. Drag the handle round to change the component's direction.



Handle for rotating component

Centre of rotation
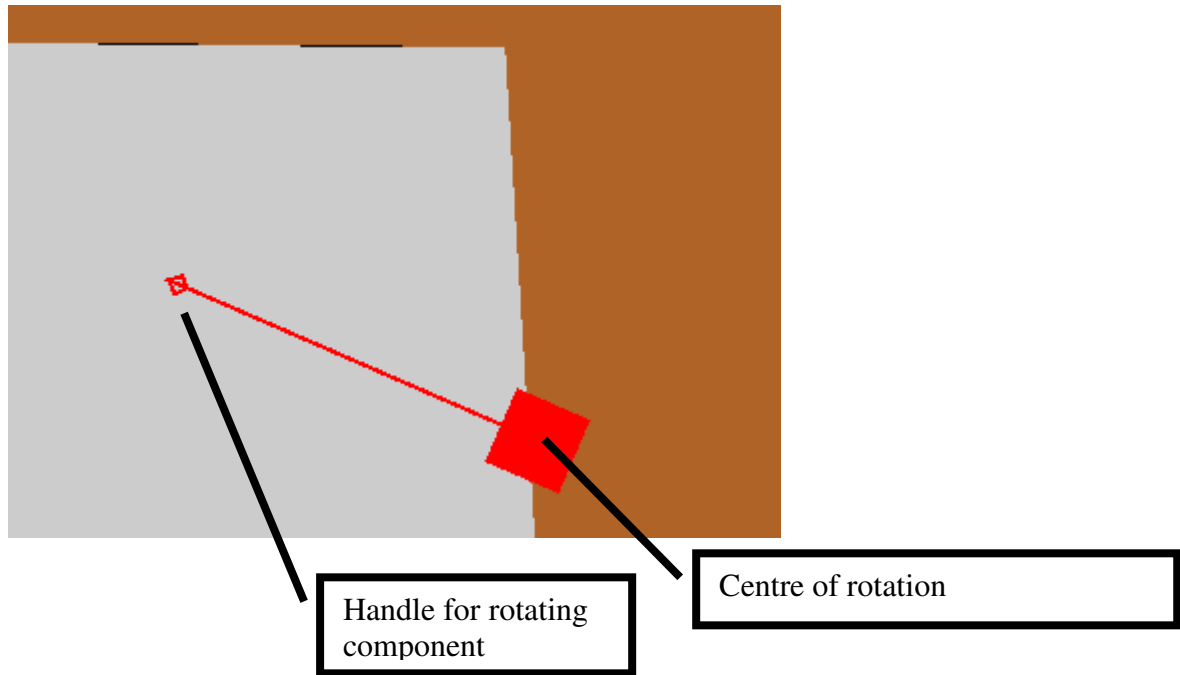
**Figure 79: Rotating handle in 2D view**

## *Warehouses and HouseBlocks*

These components are to be superseded by the Building component. They are built just like BuildingComponents. The only difference is that they use different default values for attributes such as height.

## Walls

Walls with coping stones as shown in Figure 80  are added to a jetty by doing the following:

- click on the **Add Wall** button in the jetty attribute panel
- click on a sequence of points in the 2D view where the wall is to be placed.
- click on the **Finish** button.



**Figure 80: Wall with coping stones**

A wall has distinct values for the height at the beginning of a wall and the end of a wall. Thus by building sequences of walls we can obtain effects as shown in Figure 81.
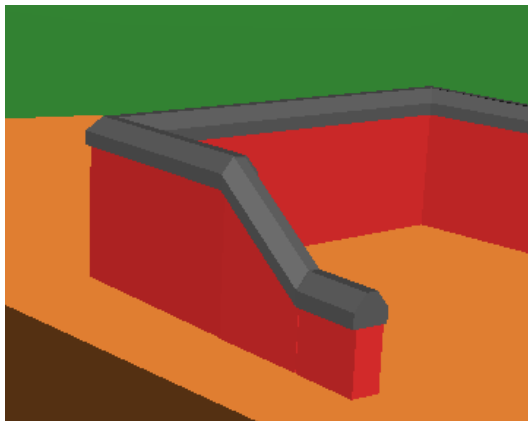


**Figure 81: Walls with changing heights**

## *Platforms*

Platforms components that can be used to build ramps to raised platforms on the jetty or to provide, for example, ramps from a jetty down into the water.

The platform has one edge, called the **ramp edge**, which can be set lower than the remaining edges. This edge is the line joining the first and last points clicked on when adding the platform in the 2D view.

For a four point platform plotted in the order P0, P1, P2, P3 the 2D lines look like Figure 73 (with the points labelled). The ramp edge is the thicker line.
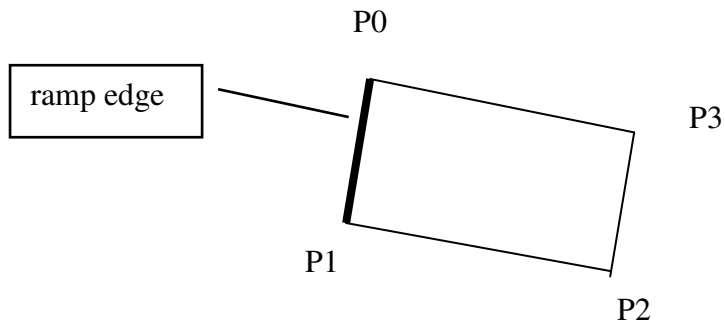


**Figure 82: Ramp edge on a 4 point platform**

In 3D this looks like Figure 83
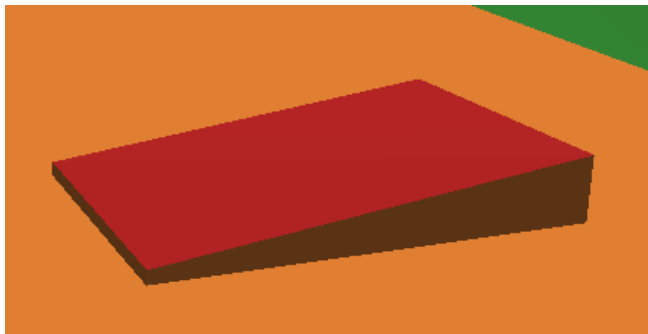


**Figure 83: 4 point platform with ramp**

For a platform with more edges, the ramp edge is connected to the adjacent points to form a sloping quadrilateral. The rest of the platform top is horizontal. Figure 84 shows a six point platform. The quadrilateral P0 P1 P4 P5 is sloping, the remainder is flat.

P5

P4

P0

P3

P1

P2
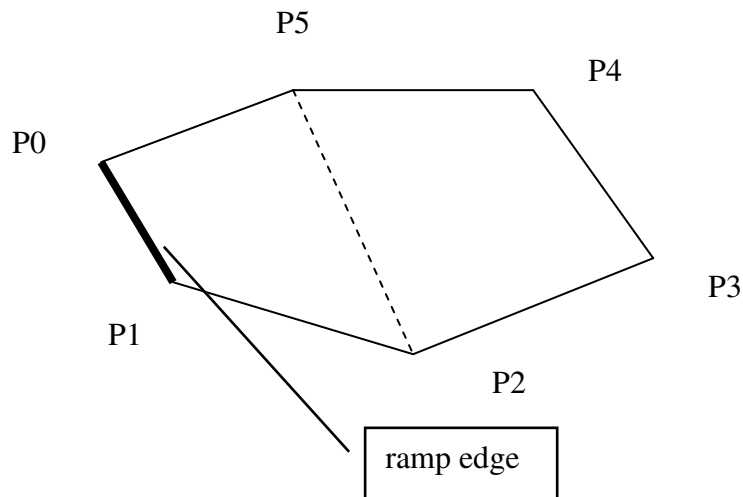
ramp edge

**Figure 84: 6 point platform with a ramp edge**

In 3D this looks like Figure 85:

**Figure 85: 6 point platform with a ramp**

## *Embankments*

The terrain provided by SRTM has quite low, 90 m resolution. Consequently the interface between land and sea/river can be quite inaccurate. Also, river banks near harbours and towns often have embankments. The Embankment component is provided as a sub-component of ScenarioScene to cater for this. Add an embankment by doing the following.

- Select the top-level node in the tree view (ScenarioScene)
- Click on the **Add Embankment** button.
- In the 2D view click on the sequence of points defining the embankment.
- Click on the **Finish** button.

The embankment attributes include:

- height above mean sea level
- colours of walls and deck

## Swing bridges

Many docks use swing bridges to provide paths for rail and road traffic that can be interrupted temporarily to enable ships to pass.

### Adding a swing bridge

A swing bridge is added to a jetty by doing the following:

- select the jetty in the 2D view or in the component tree

- click on the **Add SwingBridge** button in the jetty attribute panel

- click on the 2D view where the swing bridge pivot points is to be placed on the jetty edge. The symbol appears as a rectangle with a circle around the pivot point.

Figure 89 shows a swing bridge (brown rectangle and circle) initially placed on the 2D view..
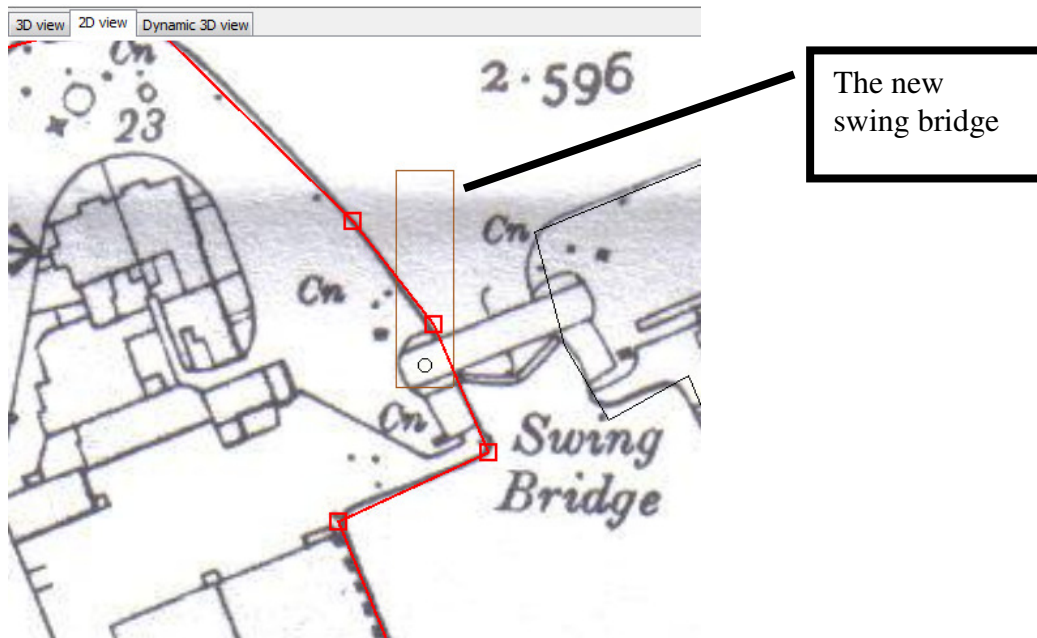


**Figure 86: Swing bridge placed on the jetty**

## Customising the swing bridge

The swing bridge's orientation, span, width and opening and closing attributes are specified as follows:

- select the swing bridge in the tree view or the 2D view

A turning handle appears in the 2D view as shown below in Figure 87:
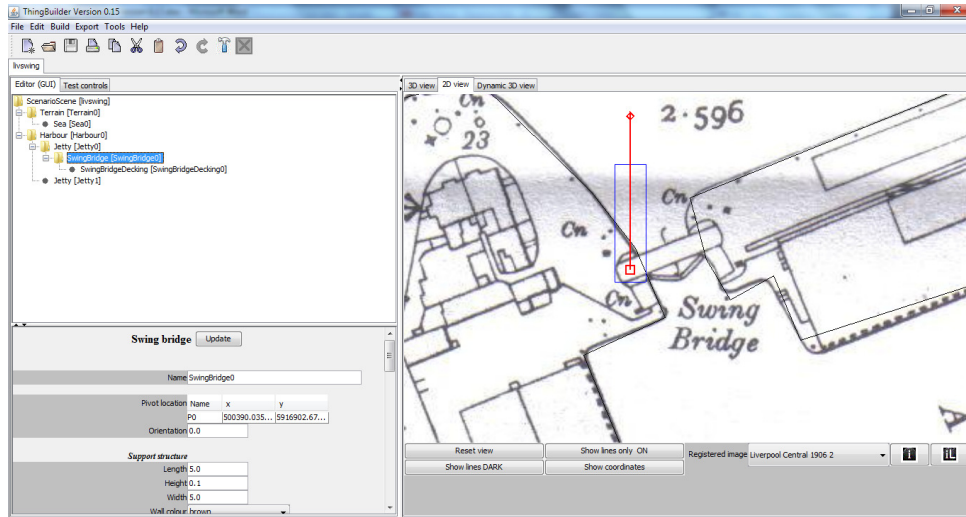


**Figure 87: Swing bridge selected in 2D view**

- finalise the swing bridge pivot point location by dragging the pivot circle into place
- use the mouse to turn the swing bridge into its correct closed position by dragging the diamond shaped turning handle. This sets the orientation attribute of the swing bridge.

The result is shown in Figure 88:
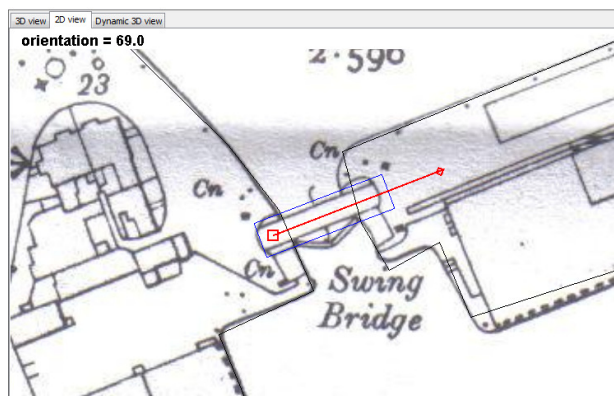


**Figure 88: Swing bridge in correct location and orientation**

- The bridge span and width can be customised by selecting the bridge's *SwingBridgeDecking* component in the tree view then modifying the roadway's *span* and *width* attributes.

The swing bridge open/closed positions are set by specifying the rotation related attributes on the SwingBridgeDecking component. The rotation angle measures the heading of the swing bridge relative to its orientation (i.e. its closed position).

To swing clockwise the rotation needs to increase towards 90.

To swing anticlockwise the rotation needs to reduce towards -90. Here are the settings for rotation need for the two directions of swing:

| | |
|---|---|
| Rotation | 0 |
| Minimum rotation | 0 |
| Maximum rotation | 90 |

| | |
|---|---|
| Rotation | 0 |
| Minimum rotation | -90 |
| Maximum rotation | 0 |
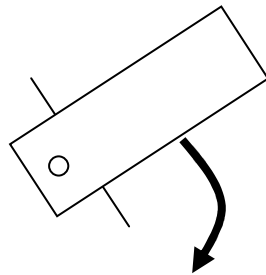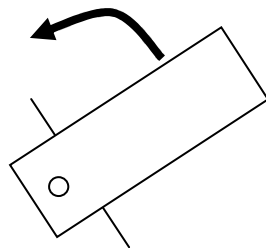
## Berths

Ships are docked in harbour at berths. Berths are added to the Scenario Scene jetties and are referred to in Scenario narrative statements such as "*HMS Hood docks at berth 123 at 06:00 am*". Ships may start the scenario in the docks. e.g. "*HMS Hood is initially berthed at berth 123.*"

A berth does not currently have a geometry. It is only visible as a symbol in the 2D view. The role of the Berth component is to identify where alongside a jetty a ship is to come to rest and in which direction. When computing the path of a ship docking, ScenarioBuilder uses the target Berth component's location and heading attributes to determine the final resting place and heading of the ship.

Berths are added to a jetty by doing the following:

- click on the **Add Berth** button in the jetty attribute panel

- click on the 2D view where the berth is to be placed on the jetty edge. The symbol appears as a thing rectangle.

- select the berth in the tree view or the 2D view

- finalise the berth location by dragging the central squre onto the jetty wall edge

- use the turning handle to finalise the berth's direction. The length of the rectangle symbol should be parallel to the jetty wall. The direction should be pointing into the water from the jetty.

Figure 89 shows a berth correctly positioned and aligned.
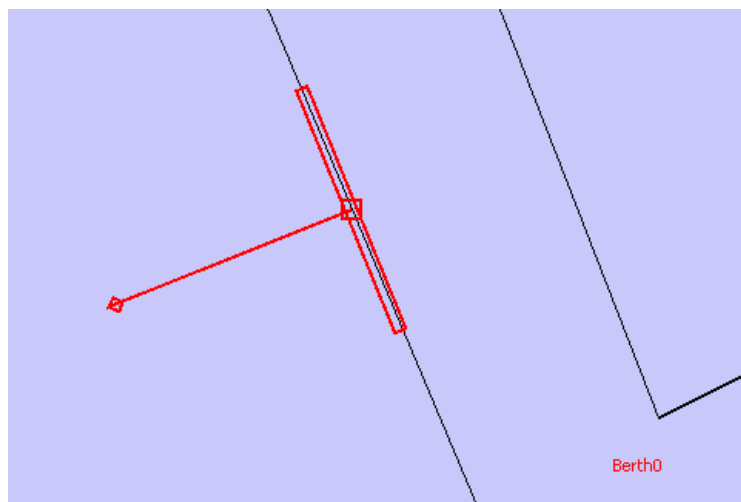


**Figure 89: Berth correctly positioned and aligned**

The next section shows how berths are linked to dockings and sailings.

## *Docking and sailing routes*

Scenarios described ships arriving in port (docking) and leaving port (sailing). ScenarioBuilder simulates these events by using components called **DockingRoutes** and **SailingRoutes**. These are paths leading to and from berths to points clear of the harbour from where movements can be described free of the limitations of the port.

In the narratives a ship can be inityially berth facing a particular way. Similarly it can dock facing a particular way.

It is intended that ships may enter and leave harbour either assisted or unassisted.

### *Assisted docking and sailing*

This makes use of tugs and other means to manouevre the ship into and out of a berth.

This has not yet been implemented

### *Unassisted docking and sailing*

The ship sails directly into or out of the berth.

This has been implemented.

## Unassisted Docking

When a narrative statement such as "*At 06:25 MayFlower docks unassisted in Liverpool Docks at berth 123*" is to be visualised, ScenarioBuilder performs the following actions:

- Find the berth with name 123 in harbour Liverpool Docks

- Find the docking routes associated with this berth that allow docking facing the required way. If none exist then report an error.

- Select one of the docking routes.

- Determine the path followed by the ship:

  o   from its previous position to the start of the docking route

  o   along the docking route taking care of turning the corners smoothly

  o   from the end of the docking route to alongside the berth at rest

  o   move sideways into the berth and come to rest

## Unassisted Sailing

When a narrative statement such as "*At 06:25 MayFlower sails unassisted.*" is to be visualised, ScenarioBuilder performs the following actions:

- Find the berth where the ship is berthed at 06:25 am

- Find the sailing routes associated with this berth for the direction the ship is facing. If none exist then report an error.

- Select one of the sailing routes.

- Determine the path followed by the ship:
  - move sideways from the berth and come to rest
  - move along the sailing route taking care of turning the corners smoothly
  - from the end of the sailing route to its next position as specified in the narrative

## Adding docking routes to a model

Docking routes are added by doing the following:

- select the harbour in the tree view

- click on the **Add DockingRoute** button

- click on a sequence of points in the 2D view where the docking route is to be placed starting with the start of the route and ending with the end of the route, near the berth.

- click on the **Finish** button.

## Hints for successful docking routes

- Make the turns as gentle as possible within the confines of the docks. ScenarioBuilder will conform to the turning circles specified for the ship in question and reports an error if the turns are too tight.

- Make the last leg of the docking route approach the berth smoothly. See figures below for good and bad examples. In the latter case the ship will run into the jetty as it attempts to turn alongside the jetty.
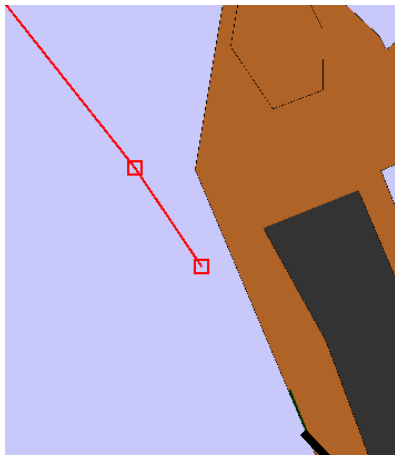


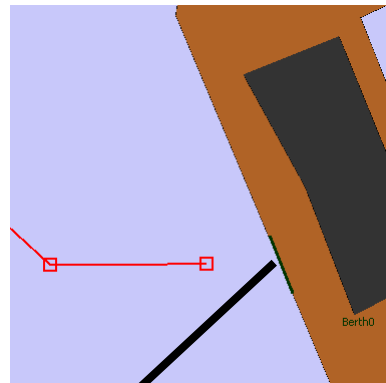**Figure 90: A good docking route end**



**Figure 91: A bad docking route end**

Berth

Note that if the ship's characteristic's prevent the geometry working out e.g. the ship cannot cope with the demands of the docking route ScenarioBuilder will report an error an attempt to suggest a solution e.g.

***Cannot reach end of move in required time with this ship's acceleration (0.3 m/s/s). Acceleration needs to be greater***

The best solution is to revisit the docking route and move the end further away from the berth to give the ship room enough to dock nicely.

## Adding sailing routes to a model

Sailing routes are added by doing the following:

- select the harbour in the tree view

- click on the **Add SailingRoute** button

- click on a sequence of points in the 2D view where the sailing route is to be placed starting with the start of the route near the berth and ending with the end of the route.

- click on the **Finish** button.

Docking and sailing routes can be adjusted like other components such as jetties and walls in the 2D view

## Hints for successful sailing routes

- Make the turns as gentle as possible within the confines of the docks. ScenarioBuilder will conform to the turning circles specified for the ship in question and reports an error if the turns are too tight.

- Make the first leg of the sailing route leave the berth smoothly. See figures below for good and bad examples. In the latter case the ship will run into the jetty as it attempts to turn alongside the jetty.
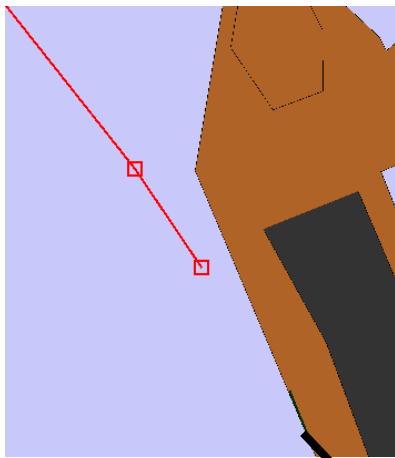

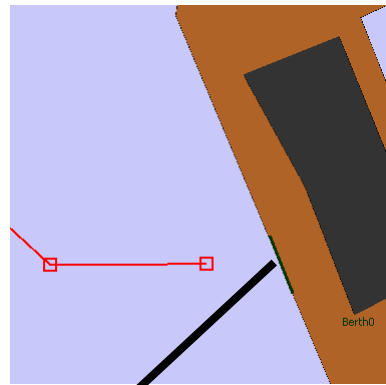
**Figure 92: A good sailing route start**



**Figure 93: A bad sailing route start**

Note that if the ship's characteristic's prevent the geometry working out e.g. the ship cannot cope with the demands of the sailing route ScenarioBuilder will report an error an attempt to suggest a solution e.g.

***Cannot reach end of move in required time with this ship's acceleration (0.3 m/s/s). Acceleration needs to be greater***

The best solution is to revisit the sailing route and move the start further away from the berth to give the ship room enough to sail nicely.

## *Adding components to the terrain*

So far we have added buildings and walls to jetties which are essentially flat bases. We can also add these components to the terrain so that they conform to the terrain. this is achieved by adding a Terrain Wall to the terrain. The points on the wall are added in the 2D view. An example is shown in Figure 94.
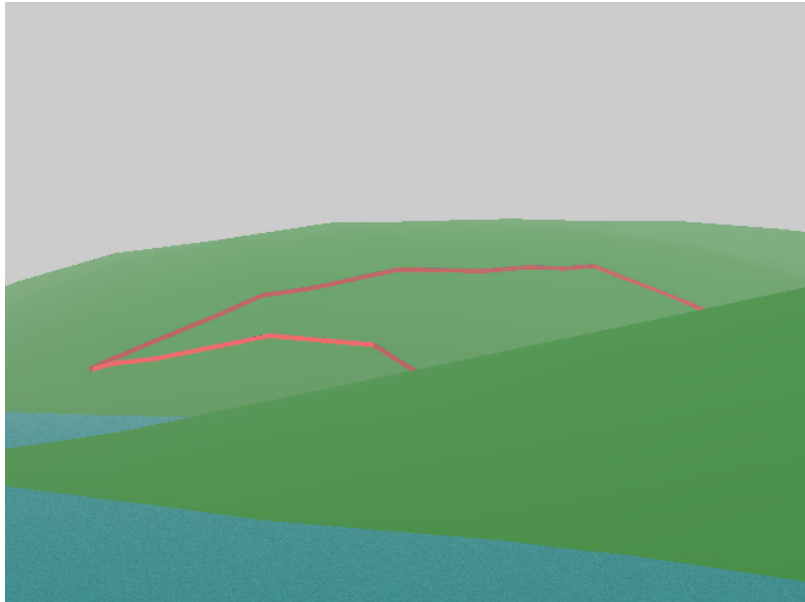


**Figure 94: A wall on the terrain**

# Viewpoints

Another kind of location on a model is the *Viewpoint*. A Viewpoint identifies a point on the model from which the scene can be viewed when the model is employed in a scenario. Its name will be listed automatically in the **ScenarioBuilder** and **ScenarioViewer** list of viewpoints.

There are two types of viewpoint:

- static viewpoints
- dynamic viewpoints

Static viewpoints are fixed in the geographical world. Dynamic viewpoints are attached to moving objects such as ships and move with the object. Thus the viewer is able to "travel" on ships.

From a viewpoint the viewer is able to turn round, look up, look down and zoom in and out.

## Static viewpoints on Scenario Scene models

Static Viewpoints are currently available on scenario scene objects. They are attached scenario scene by doing the following

- Select the scenario scene itself (top node) in the tree view
- Click on the button **Add Viewpoint**

Figure 96 shows a newly added scenario scene viewpoint selected.
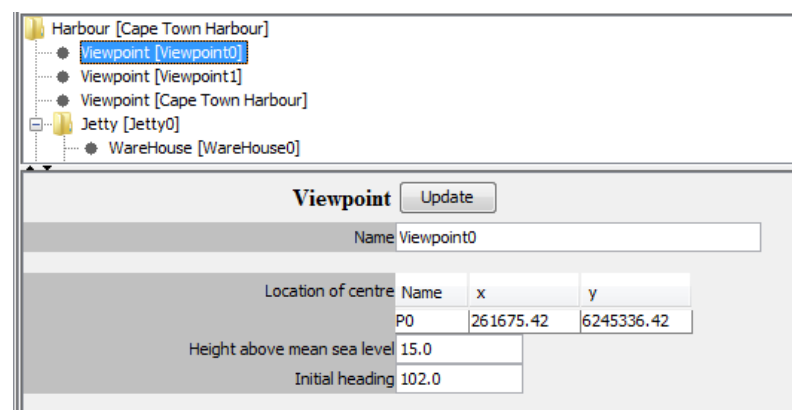


**Figure 95: New viewpoint added to a harbour**

It has four attributes:

| Name | The name that will appear in the ScenarioBuilder list of viewpoints |
|---|---|
| *Location of centre* | The location is expressed in UTM (Universal Transverse Mercator) format. |
| *Height above sea level* | This is measured in meters. |
| *Initial heading* | Initial direction in which viewer is facing |

## *Dynamic viewpoints on ship models*

Dynamic Viewpoints are currently available on ship objects. They are attached to the ship's hull by doing the following

- Select the ship's hull in the tree view

- Click on the button **Add Viewpoint**

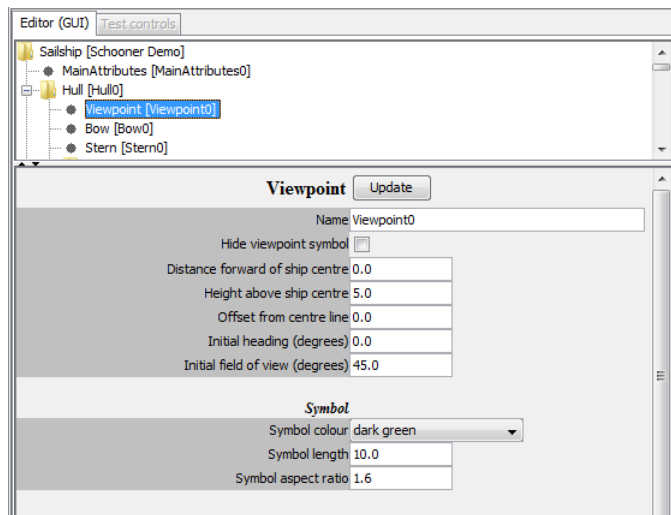Figure 96 shows a newly added viewpoint selected.



**Figure 96: New viewpoint added to a ship's hull**

There are several attributes

| | |
|---|---|
| *Name* | The name that will appear in the ScenarioBuilder list of viewpoints |
| *Hide viewpoint symbol* | Used to show/hide the symbol in the 3D view while editing the model. When exporting to ScenarioBuilder, this should be hidden. |
| *Distance forward of ship centre* | The ship centre is specified in the top level node in the tree view. |
| *Height above ship centre* | |
| *Offset from centre line* | Starboard is positive, port is negative |
| *Initial heading (degrees)* | This heading is realtive to the bows of the ship. Thus 0 degrees will be looking directly forward, 90 degrees will be looking directly starboard, 180 degrees will be looking directly aft etc. |
| *Initial field of view (degrees)* | This affects the zoom. The smaller the value, the greater the zoom effect. |
| *Symbol colour* | The colour used when displayed in 3D while editing the model |
| *Symbol length* | The symbol is a pyramid with point at the viewpoint and base facing the direction the viewer is looking. |

| Symbol aspect ratio | The aspect ratio of the base of the symbol pyramid. |
|---|---|

Figure 97 shows a Viewpoint made visible in the editor (blue because it is selected).
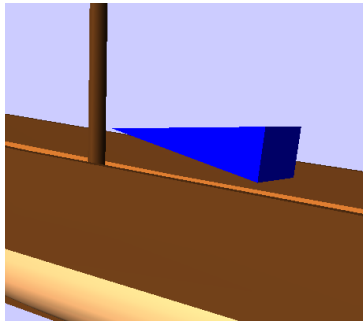


**Figure 97: Selected Viewpoint on a ship**

The viewer will be at the pointed end and the direction of viewing will be towards the wide end.

## Naming viewpoints

In order to organise the viewpoints for the different ships etc, ScenarioBuilder lists them with the ship name unindented followed by the indfented viewpoints for that ship as shown in Figure 98.
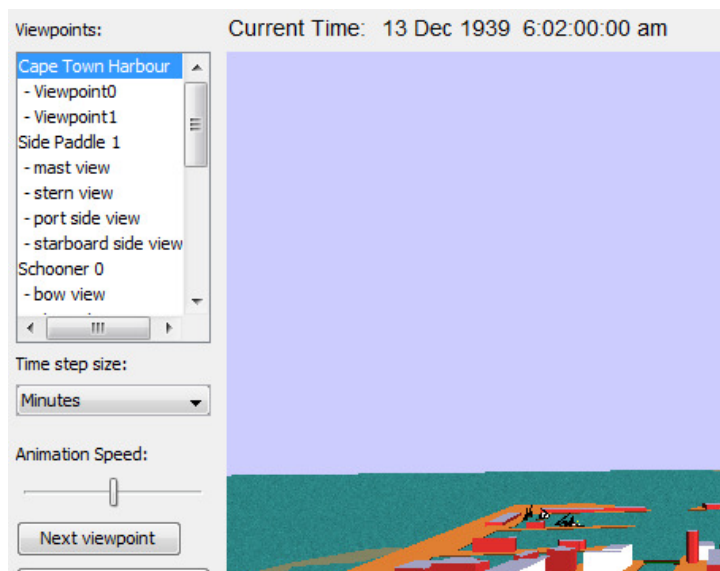


**Figure 98: Viewpoint list showing indentation**

Note that Cape Town Harbour is the unindented title viewpoint while Viewpoint0 and Viewpoint1 are indented.

If a ship is not given a viewpoint then its name will be given to a default viewpoint around the centre of the model. If you add a viewpoint with the name of the ship then that will be used for the unindented, title viewpoint instead.