Ozlab Guide

June 30

2019

Ozlab is a GUI experimental web service of Karlstad University based on the Wizard-of-Oz concept. This technique is used to test a variety of ideas for interaction design without programming the interactivity. Instead there is a human, a "Wizard", who manages the user interface in real time. This guide gives an overview of the main functions of the Ozlab system and should facilitate for the beginner to do interactive demonstrations and usability experiments by this web service.

Quick Guide: Short commands in the Shell Builder

Ctrl/cmd + S	= Save
Ctrl/cmd + A	= Select all objects on a scene
Ctrl/cmd + C	= Copy
Ctrl/cmd + V	= Paste
Ctrl/cmd + left click	= Selects multiple objects (one by one)
Delete	= Deletes all selected objects on a scene

(Ctrl/cmd + Z has not been implemented yet)

Table of contents

1. Introduction

Ozlab runs best in the web browser *Google Chrome*. Some functionality might not work in for example *Firefox* or *Safari* due to the browsers' technical differences.

Shells, or interaction shells, are the Ozlab prototypes. In a shell, different appearances ("Scenes") can be created. They are called shells since they are not stand-alone prototypes – the shell will not function if no Wizard is present.

Ozlab supports experimentation by the Wizard-of-Oz method, i.e., where interaction designs are demonstrated without programming. Instead there is a human, a "Wizard", who manages the user interface in real time. In its present version, Ozlab supports three roles in Test Runner mode, namely the Wizard, a "user" of the pretended system and observers (see Figure 1). There is also a preliminary phase where the designers put one or several interaction shells together (graphics, texts, links to external content). Thus, there are two main modes in which Ozlab can run, Test Runner mode and Shell Builder mode.

As Ozlab functions like a web service, the web browser must be given a web address. This provides an access to the Ozlab server. To connect to Ozlab it is necessary to go via the network where Ozlab is hosted (at Karlstad University it is one of KAU's networks).

An interaction designer may use one and the same address to build shells and to run tests on. When running a test session, the web browser of the Test Participant also uses this web address and will then automatically be connected to the current shell. This is also how it works for observers who use Ozlab to view the test session.

Quick Guide: Icons used in the Ozlab Guide	
✿	= Instructions
♀	= Good to know
❶	= Note
✨	= Tip

2. The landing page of the Ozlab system

When accessing Ozlab in a web browser, the user ends up at a landing page from which one can access the three different roles in Ozlab: Test Leader, Test Participant and Test Viewer (see Figure 1).

Besides these roles, there are two modes shown in Figure 2. The "Ozlab Test Runner" column on the left side pertains to the Test Runner mode, while the "Ozlab Shell Builder" column to the right is used to access the Shell Builder mode.

2.1. Roles in Ozlab

TL – Test Leader
The Test Leader builds the interaction shell in the Shell Builder mode and then acts as Wizard to control the interaction shell in Test Runner mode.

TP – Test Participant/Test Person
Note that the Test Participant might be a team mate, etc.

TV – Test Viewer
The Test Viewer can see the Test Participant's view but cannot affect the interaction or the interaction shell.
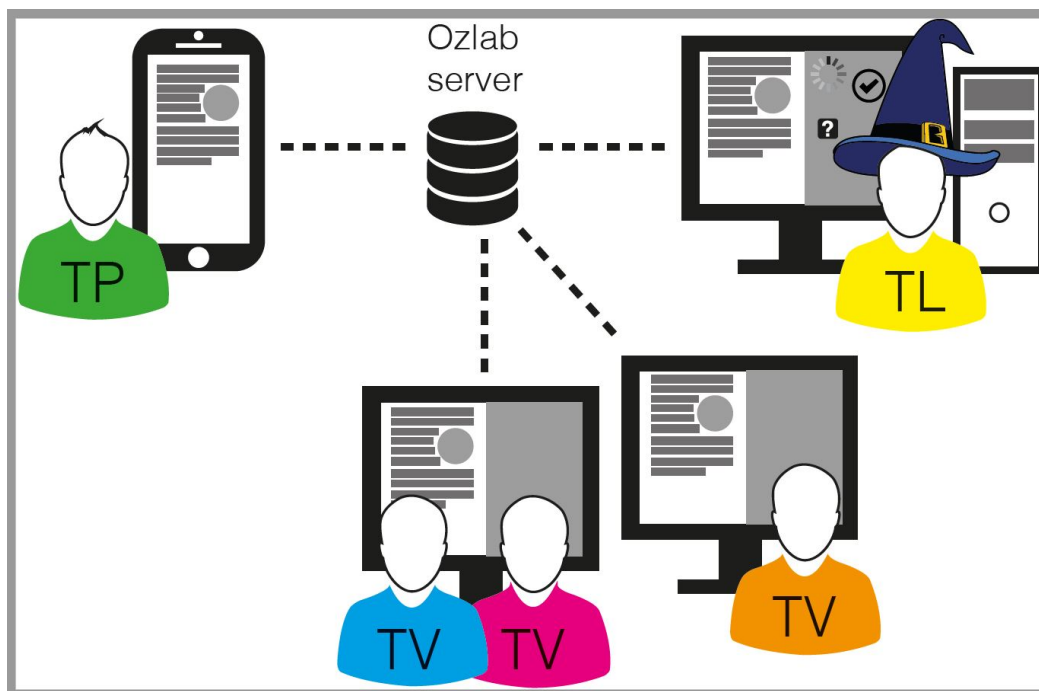
Figure 1. Roles during a test session

2.2.   The "Ozlab Test Runner" column

Ozlab Test Runner is for demonstrations, tests or checking an interaction shell.

⚙ **Test Leader (Wizard)**: Clicking on "Start new session" opens the Test Runner directly if an existing shell is chosen under "Please select shell". The first scene of the shell will be opened if another scene is not picked.

⚙ **Start TP Window:** Clicking on "Join test session" opens the Test Participant's view of the interaction shell.

This view should be opened on the device or computer that the Test Participant is going to use. If the Wizard has not yet started a test session, this view will be a black background with a white spinning wheel.

⚙ **Test Viewer:** Clicking on "View on-going session" opens the Test Viewer mode of the Test Runner, where the interaction between the TL and the TP can be observed, but not affected.

2.3.   The "Ozlab Shell Builder" column

Shell Builder is for building new shells or editing, importing or exporting existing shells. There is no differentiation as to roles in this mode and there is only one box in this column.

⚙ **Build or edit shell:** Clicking this button opens the Shell Builder.
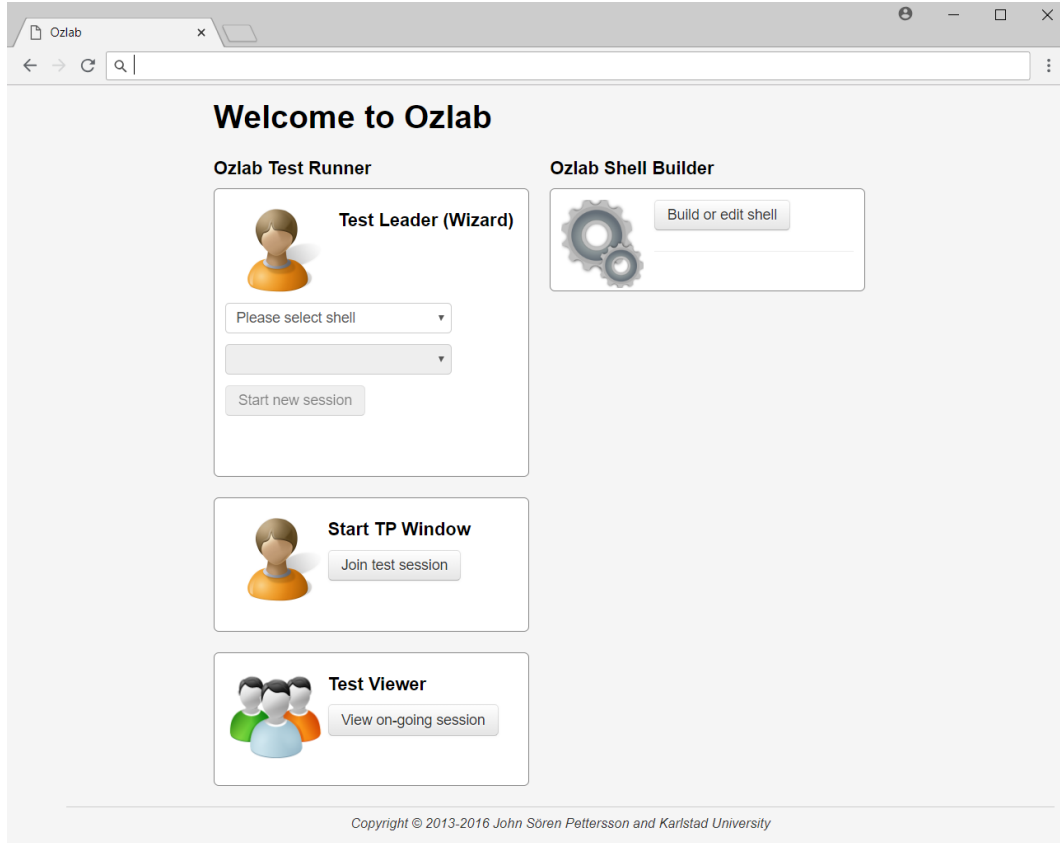
Figure 2. Overview of the different states of Ozlab

3. Shell Builder

Ozlab Shell Builder is a main part of the Ozlab system. It is used for designing and editing interaction shells, that later can be used for e.g. usability tests or evaluations.

3.1. Structure of Shell Builder

In Figure 3 the Graphical User Interface (GUI) of the Ozlab system in Shell Builder mode is shown.

In the upper left corner of the GUI, "File" can be clicked to open a shell. In the upper right corner, it is shown whether a TP or a TV is connected.

The Shell Builder mode is separated into the "General Panel", the "Work Area" (including "Active Scene") and the "Settings Panel".

The General Panel has four different categories (panes). The first category lists all "Scenes" of the shell. In the next category, all usable "Objects" are listed. The third and fourth category could include manually added "shell objects" and "scene objects".

The Work Area is the working surface for the Wizard. It includes the Active Scene as well as the grey area around. The TP and TV will only see the Active Scene (white area) during a session.

The Settings Panel is on the right side of the system. The first category of the Settings Panel is "Object Properties". Here, features of objects can be changed. In the second category, "Object Behaviors", several behaviours can be added to the objects.

At the bottom, there is a "Log" function to track all actions during a session. The logs can be shown by clicking the arrow at the lower right.
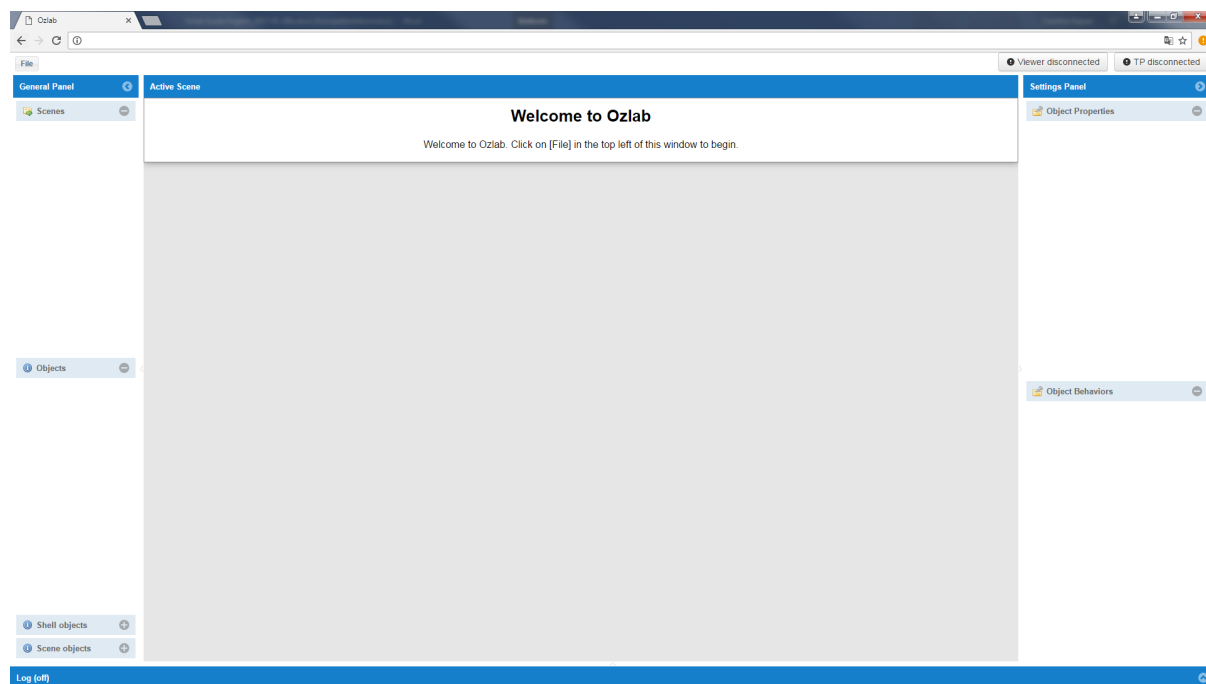
Figure 3. Ozlab Shell Builder (without an Active Scene in the grey Work Area)

3.2. Shells

Create new shell

⚙ To create a new interaction shell, select "File" (see Figure 4) and then "Create shell". Write the name of the interaction shell in the appearing dialog box and click "OK".
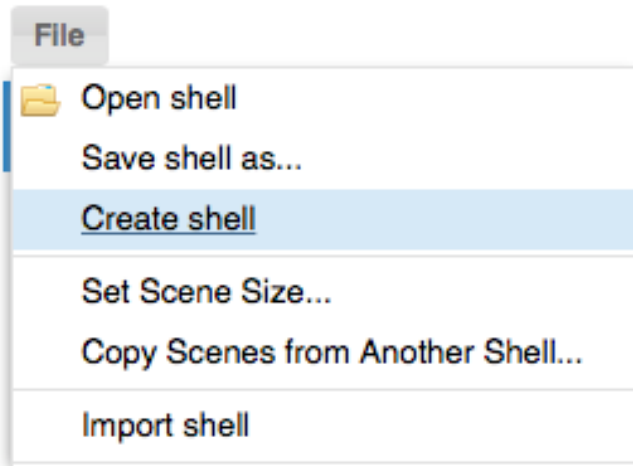
Figure 4. Creating a new shell

Open existing shell

⚙ To open an existing shell, select "File" then "Open shell". Select the desired shell in the list and click "Open". Figure 5 shows this dialog.
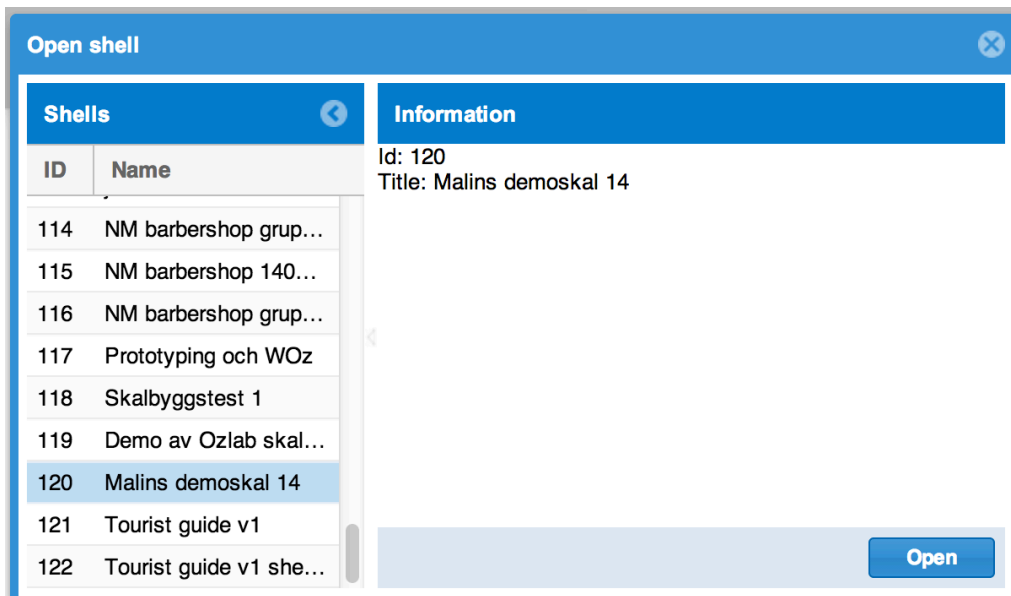
Figure 5. Opening an existing shell

Import existing shell

⚙ To import a shell created outside of the online Ozlab environment, select "File" then "Import shell". Locate the file on your computer and upload it.

Exporting shell

To export an interaction shell, select "File" and then "Export current shell" (see Figure 6).
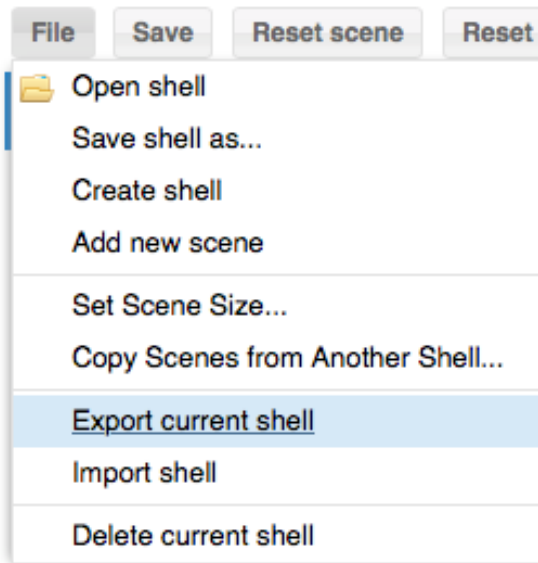
Figure 6. Exporting a shell

3.3. Scenes

Scenes are the pages of the interaction shell the TL can use to build up different appearances in the shell, which the TP then can interact with.

Work Area

The Work Area includes Active Scene with a white background as well as the grey area around. Objects placed outside of the Active Scene will not be visible for the Test Participant, but can be used as an object storage by the Wizard during the test. Just remember to give the object a movable behaviour (ObjectMovableByTL).

Templates

To avoid having to make the same changes on several scenes, such as add background and structures that will appear in many scenes, a scene with the general objects can be created and then copied within the same interaction shell.

The template scenes and the specific scenes of an interaction shell can also be divided by using different shells. At any time, scenes from another shell can be imported into the interaction shell by using "Import Shell" in the "File" menu, and then delete unwanted scenes.

Note that all scenes from the other shell will be imported.

Add new scenes

Click on the button "Add New Scene" in "Scenes" of the "General Panel". Write what the scene will be called and click "OK".

It is helpful to name the scenes to something that makes TL understand what the scene contains, this will make not only designing the interaction shell easier, but running it as well.

Copy existing scenes

Click the gear behind the scene name in the General Panel on the left side and click "Copy" in the dialog box (see Figure 7). If necessary, replace the name of the copied scene (which is now called "The scene name_Copy").
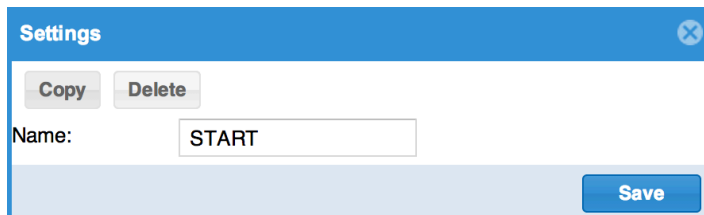
Figure 7. Copying and deleting existing scenes

Tip: If any/many scenes in the shell should have the same background or the same object in some places, create a template and copy it. Adding background and/or objects several times is not necessary then.

Delete existing scenes

Scenes that are not required in the shell can be removed.

To delete a scene, select it and click on the gears after the scene name. Click "Delete" and then "Yes" in the dialog box (see Figure 7).

Changing Active Scene size

The Active Scene size determines the size of the surface the TP will see. There are several predefined scalable scene sizes that can be selected depending on which device the TP has.

To set the Active Scene size, select "File" then "Set Scene Size". Select the desired size and click "Save" (see Figure 8).

The Active Scene size is constant throughout the shell. If the shell shall fill the entire screen of the TP device, the size of the TP device must be entered. Areas outside the Active Scene will appear as a grey area.
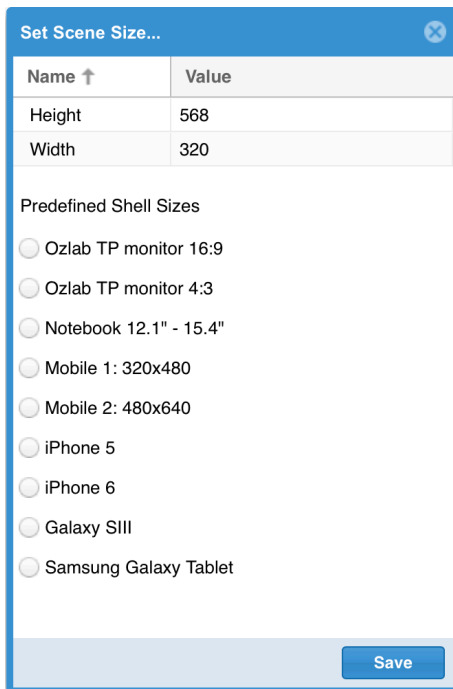
Figure 8. Several available scene Sizes

Import scenes from a different interaction shell

In the Shell Builder objects and scenes can be imported from other shells. All scenes and objects are imported from the other shell. It is the current open shell that determines the Active Scene size.

To import an interaction shell, select "File" and then "Import shell". Select the desired shell by clicking "Browse…" in the appearing dialog box (see Figure 9). Then enter a shell name and click "Import".
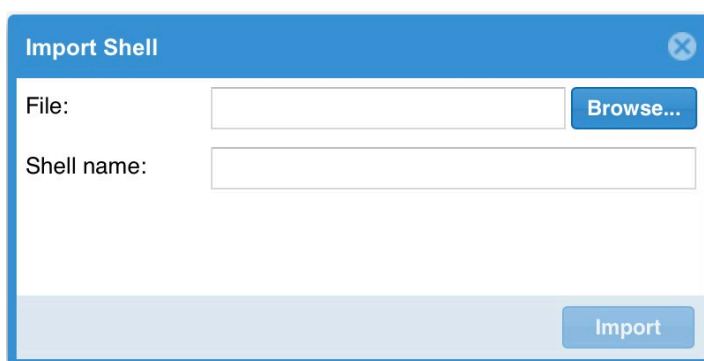
Figure 9. Import Shell

3.4. Objects

There are several different generic items (objects) to choose from in the Shell Builder as shown in Figure 10(a). The objects are used for making something to appear in the interaction shell. A shell can be composed of e.g. an image and text objects, or a combination of graphics and other types of objects. Objects can be added to the Active Scene by using drag and drop. Objects are edited in the Settings Panel which contains two parts, the Object Properties and Object Behaviours, respectively; see Figure 10(b).
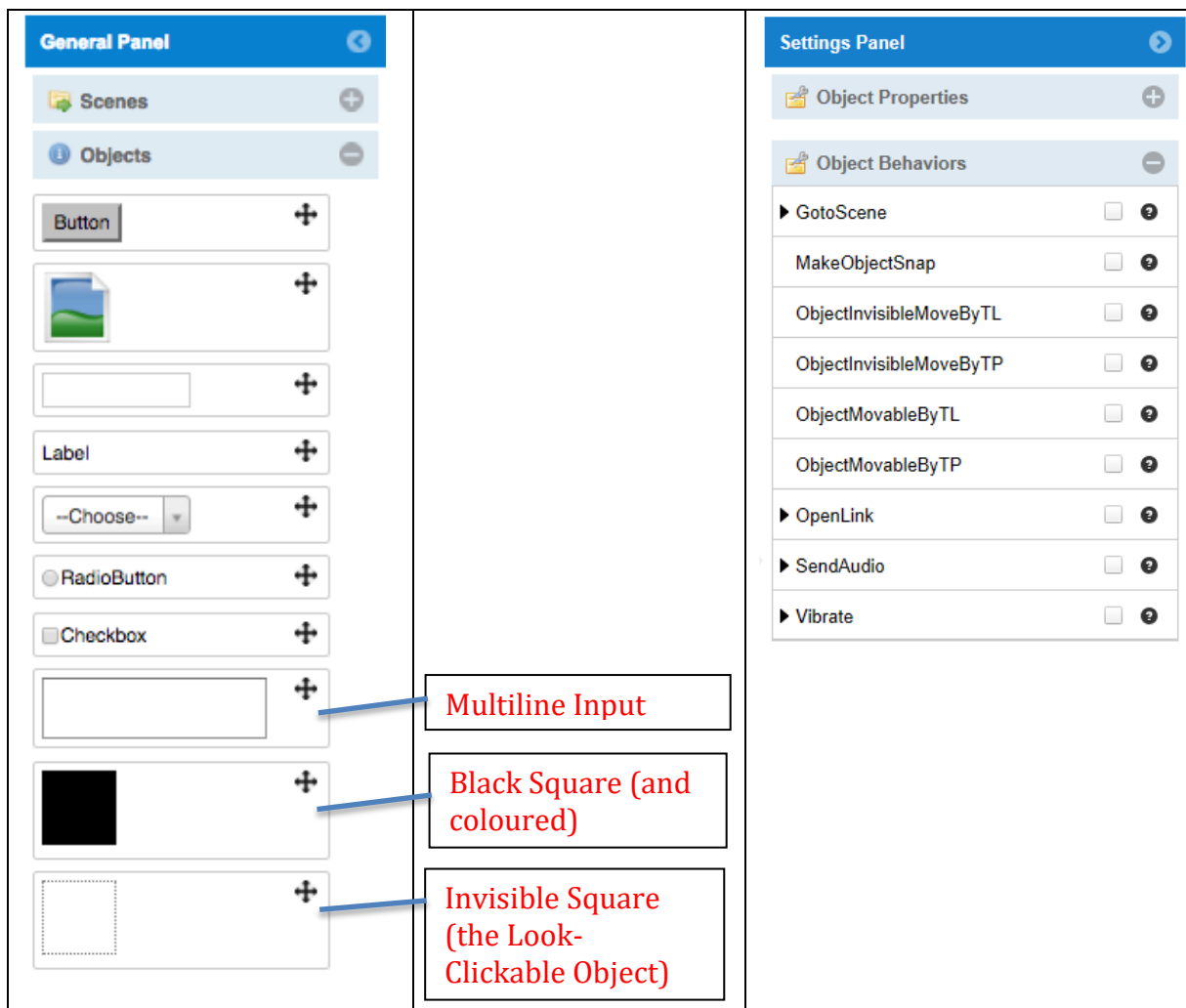
Figure 10. (a) The different generic items in the General Panel, (b) The Settings Panel for Objects

Shell objects

If adding an object, e.g. an image, to the Shell objects pane (to the left in Figure 3), the object with all its settings and features can be dragged and dropped several times from the Shell objects pane to the Work Area. Figure 12 shows a close-up where there are two objects in the pane (and one in the Scene objects pane).

"Shell objects" are available on every scene of an interaction shell. By using shell objects duplications of effort can be avoided as the shell objects have the same features in every scene of the shell.

Scene objects

If adding an object, e.g. an image, to the Scene objects pane, the object with all its settings and features can be dragged and dropped several times from the Scene objects pane to the Work Area.

"Scene objects" are only available in one Work Area of the interaction shell. By using scene objects duplications of effort within a scene can be avoided as the scene objects have the same features in a particular scene.

When copying a scene, the scene objects will also be available in the new scene.

Adding objects to Shell objects pane or Scene objects pane

Right-click the item on the Active Scene and select "Add object to Shell objects" (see Figure 11) or "Add object to Scene objects".
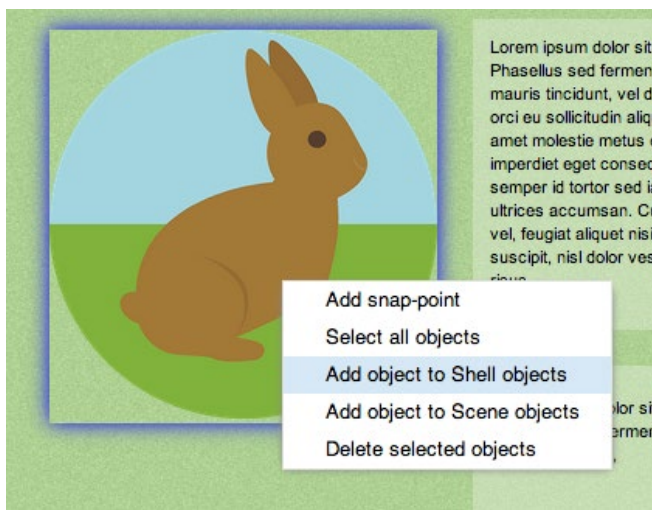
Figure 11. Adding an object to Shell objects

Button

Looks and functions like a normal button. The button changes its appearance depending on which operating system (e.g. Windows or OSX) the TP device is running. Button text (value) can be replaced and the height/width can be set.

Image

Used to add pictures/images/photos/scanned sketches to the interaction shell. If no changes are made to the size settings (height and/or width), the image is displayed in its original size. Change height or width to keep the image's proportions, or drag the image object's edges to resize the image (hold down "Shift" to maintain the proportions).

Tip: If many and heavy pictures will be used in the shell, the images can take a long time to load. Scale large images in an image editor instead of setting the size in Ozlab.

Input field

Creates an input field in which the TL and the TP can type one line. Height and width can be set, but the input field is always single-row regardless of height. (For multiple-line input fields, see "Multiline Input field".)

Values (text) that is fed into the field can be displayed on the same or another scene; see section on Hidden fields.

Label

Labels are used mainly for text. There are several settings that can be made, e.g. fonts, text size, colour, background colour, and the size of the container (width and height). Resize the label container size by dragging the edge (in the same way as image object).

Under the "Source editor" certain code can be inserted into the label. This can be used to embed iframes, such as a film (e.g. from YouTube) or a map (e.g. from Google Maps).

Note that the TL will not be able to follow what the TP does in an iframe.

The Source editor can also be used to add custom HTML and CSS.

The label object can be used to showcase the values of other items; see section on Hidden fields.

Drop-down menu

Looks and works like a regular drop-down menu. Options can be added and sorted. The width of the menu can be changed. Values of the drop-down menus can be displayed on the same or another scene; see section on Hidden fields.

Radio Button

Used when only one choice is possible. Options can be added and sorted. The list can be displayed vertically or horizontally by selecting the "Inline" checkbox in the Object Properties pane. The margin between the list items can be changed. Values of radio buttons can be displayed on the same or another scene; see section on Hidden fields.

Checkbox

Used when the TL or TP will be able to tick several choices. Options can be added and sorted. The list can be displayed vertically or horizontally by selecting the "Inline" checkbox in the Object Properties pane. The margin between the list items can be changed. Values of checkboxes can be displayed on the same or another scene; see section on Hidden fields.

Multiline Input field

Creates an input field in which the TL and the TP can type in. Height and width can be set, but if TP writes a long text the field will get a scroll bar. There is also the possibility to set a maximum amount of characters that TP can type in the field by editing "Length (max)". After entering the number, one has to also click in the Object Properties pane just to confirm the new setting. A zero in the length field means unlimited text input.

Values (text) that is fed into the field can be displayed on the same or another scene; see section on Hidden fields.

Black Square (and coloured)

The black, or in fact *coloured*, square can be used to represent various rectangular elements without the need to create specific images and use the Image-object. The height and width of the square can be changed in order to have a thin line in a grid or a

table, or as a border or background colour. The colour of the square is changed by the palette in the "Object Properties" menu on the right side of the screen.

Invisible Square (the "Look-Clickable" object)

This square can be added on top of a static image and reshaped into a rectangular form. When the TP hovers this part of the image, the mouse pointer will change into the common "pointing hand".

By adding the Invisible Square to different parts of a scene with a mockup of a webpage, these parts will appear as "real" links for the test person.

The wizard navigates to the linked page. It is also possible to add a real link (OpenLink in 3.6.) to the Look-Clickable object or to use the function "GotoScene" (3.6.).

Look-clickables can be used when one takes a screen shot of a web page and wants a quick way to re-use the links found in the screen shot. By adding the look-clickable to the different links on the web page they will appear as "real" links to the test person. Of course, it can also be used to quickly make a rip-off of the windows of an application.

3.5. Object Properties

In the Ozlab system, properties of every object can be changed in the Settings Panel on the right side of the screen (Figure 10b).

For the settings, Ozlab differs between two types of objects: Objects in the General Panel (Shell objects pane or Scene objects pane) and Objects in the Work Area.

Objects in the Shell objects pane and Scene objects pane

By making changes in the shell objects settings or scene objects settings, duplications of effort can be avoided afterwards.

To change settings of objects in the panes with Shell objects and Scene objects click the gear next to the object in Shell objects pane or Scene objects pane in the General Panel (see Figure 12). On the right side in the Settings Panel changeable Object Properties (see Figure 13) will appear.
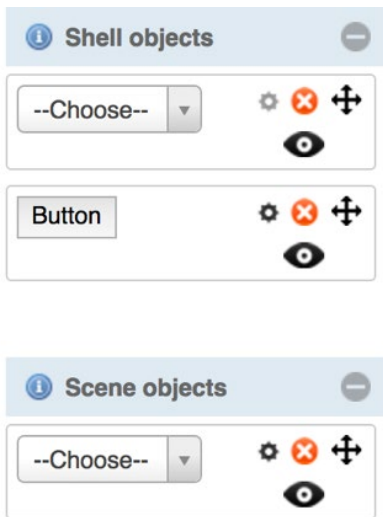
Figure 12. Shell objects and Scene objects

Note that changing the shell objects settings or scene objects settings will not affect already existing objects on the scenes of the shell. Only new objects which are added to the Work Area will have that settings.

Objects in the Work Area

Settings of specific objects in the Work Area can be changed.

Click on an object to invoke the specific settings on the right side in the "Settings Panel" (see Figure 13).

Note that changes here will only affect the selected object in the Work Area.
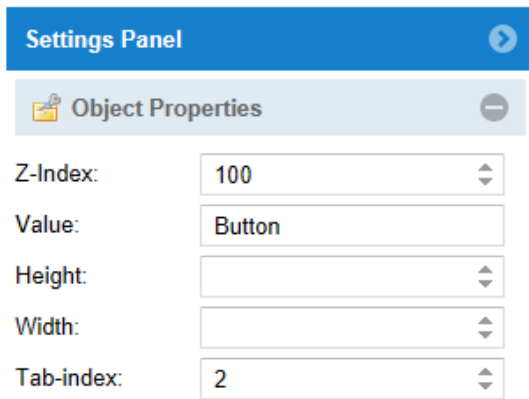
Figure 13. Object Properties

3.6. Object Behaviors

There are several behaviours in the Ozlab system that can be added to objects. The behaviours are active in Test Runner mode (i.e. to use / try the behaviours Ozlab must be in Test mode). In the user interface, the American spelling "Behaviors" is used.

For the behaviours, Ozlab differs between two types of objects: objects in the Shell objects pane or Scene objects pane in the General Panel and objects in the Work Area.

Objects in the Shell objects pane and Scene objects pane

By adding behaviours to objects in the Shell objects pane or Scene objects pane, duplications of effort can be avoided afterwards. The added behaviour is saved for the shell object or scene object. Every new shell object or scene object which is added to the Work Area will have that behaviour.

To add behaviours to objects in the panes with Shell objects and Scene objects click the gear next to the object in "Shell objects" or "Scene objects" (see Figure 12). On the right side in the Settings Panel addable "Object Behaviors" (see Figure 14) will appear.
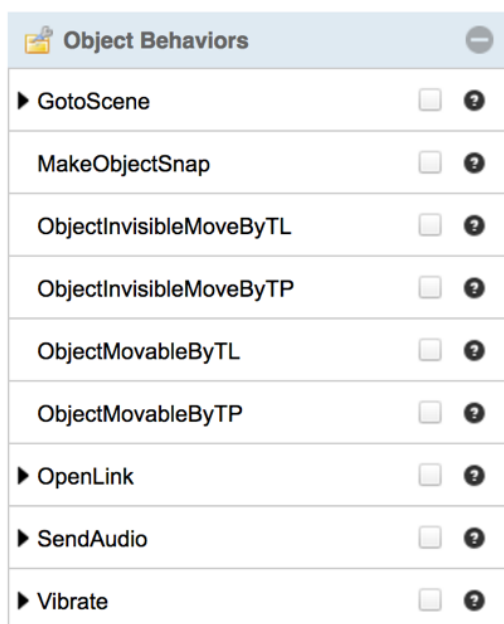
Figure 14. Object Behaviors

Different behaviours can be added by ticking the box next to the behaviour. It is necessary to extend some behaviours by clicking the small arrow in front of the title (i.e. choosing the scene for "GotoScene").

Note that adding behaviours to objects in the Shell objects pane or Scene objects pane will not affect already existing objects on the scenes of the shell. Only new objects added to a scene will have these behaviours.

Objects in the Active Scene

To add behaviours to objects on the Active Scene, click on a specific object. On the right side in the Settings Panel addable "Object Behaviors" (see Figure 14) will appear.

Different behaviours can be selected by ticking the box next to the behaviour. It is necessary to extend some behaviours by clicking the small arrow in front of the behaviour name (i.e. choosing the Scene for "GotoScene").

Note that added behaviours will only affect the selected object in the Active Scene.

GotoScene

Makes the object a scene link. When the TL or TP clicks the objects, the desired scene will be shown.

To select what scene the object should link to, add the behaviour, extend it and enter the scene name (see Figure 15).

Be careful with the spelling of the scene name. If a linked scene is not found, an error message will appear in the TL view, and no scene will automatically show up.

Figure 15. GotoScene

MakeObjectSnap

To understand this part, read "Snap Points" on p.19 first.

Makes the object centre over the "snap points" in Test Runner mode if a drag-and-drop ends over such a point.

Note that this behaviour must be added last of all behaviours (e.g. after ObjectMovableByTL/TP) to the objects in order to work. Objects pulled from the Shell objects pane or Scene objects pane must first be released on or off the Active SceneWork Area to be centred over a snap point.

ObjectInvisibleMoveByTL

Makes the item movable for the TL. The movement cannot be seen by the TP.

Only when TL drops the object, the object's new position is visible to the TP.

ObjectInvisibleMoveByTP

Makes the item movable for the TP. The movement cannot be seen by the TL.

Only when TP drops the object, the object's new position is visible to the TL.

ObjectMovableByTL

Makes the item movable for the TL. The movement can be seen by the TP.

ObjectMovableByTP

Makes the item movable for the TP. The movement can be seen by the TL.

If an object is moved and dropped on a bigger object with a higher z-index, the dropped object will disappear. Make sure to change to the correct z-index on all movable objects. You can read more about "Z-index" on p.18.

OpenLink

The object becomes a link to an external site. To select what external site the object should link to, add the behaviour, extend it and enter an URL.

Remember that when the TP opens a link to an external site the TL loses the control over what the TP does. The TL cannot follow the TP's mouse movements, or see what the TP navigates to.

SendAudio

A sound is played when the TL or the TP clicks on the object. One should turn off the audio on the TL's device when doing GUI-ii (section 7.3) or other experimentations where the TP can hear the TL (and the TL device). To add the behaviour, select and extend the SendAudio in Objects Behaviors, click "Browse…" and choose a sound file.

Wizards prefer short sound snippets of a few seconds duration only as they are unable to pause or stop the sound once it starts to play.

If one clicks on the object with a SendAudio behaviour several times, multiple tracks with the same sound will be played on top of each other.

Note that it only works on Android devices. Sound must be enabled on the device for the behaviour to work.

Vibrate

Makes the object to send a vibration signal. Should only be controlled by TL (if TP has an object with the vibration behaviour, only TL's device will vibrate (if TL is using a mobile device).

To adjust the duration of the vibration, adjust the value of VibrationLength accordingly.

This only works on Android devices and vibration must be enabled on the device.

3.7. Other features of the Shell Builder

Hidden fields: Save and display values

Values from the drop-down menus, input fields, radio buttons and check boxes can be saved in a hidden field (a container). The value can then be printed in another place on the same scene or another scene. This functionality could be useful if what the TP has written/selected should/needs to appear at another place, e.g. if the interaction shell contains an order form and an order summary. The functionality can also be used by the TL, for example to add text or embedded material during a test session. Click the gear on the object whose value is to be saved, for example, an input field.

1. Check the "Save value to field" and name the hidden field below, e.g. to "Address" (see Figure 16).
2. Go to the scene where the value should be printed and select a Label object.
3. Click on the gear of the Label object and check the "Get value from field" (see Figure 17).
4. Type in the hidden field name (such as "Address") in the "Get value from field".
5. If the value should be displayed on the same scene as from where it was retrieved, the scene must be updated. The easiest way is to use the "Show wait screen" or the "Freeze" function.

Figure 16. Saving values (I)

Figure 17. Saving values (II)

Select, delete, copy, move objects

Select multiple items: Ctrl/Cmd-clicking on the objects. Some items, such as dropdown menu, must be selected by clicking on the toolbox.

Select all items: Ctrl/cmd + a.

Delete objects: Right-click and select "Delete selected objects".

Move objects: Drag one of the selected items, or use the arrow keys.

Copy objects: Ctrl/cmd + c.

Deselect items: click a blank space in the Work Area.

Z-index

It is possible to put objects in front of other objects. The Z-index determines which objects should be in the front and which one in the back. The object with the highest index will be in the front.

Z-index is set to objects by clicking either the gear icon behind the object in the Shell objects pane or Scene objects pane in the General Panel or by clicking the object in the Work Area in case of a specific object. The Z-index can then be set on the right side in "Object Properties" in the Settings Panel.

To avoid having to set the Z-index on many objects, set the correct Z-index of the object in the panes Shell objects or Scene objects before the object is placed on the Work Area.

Set the Z-index from 0 and up (in the range of the integer, see Figure 18).

Minus values makes the object settle behind the Active Scene, which means that the object becomes impossible to obtain. The Default Value for the Z-Index is 100.
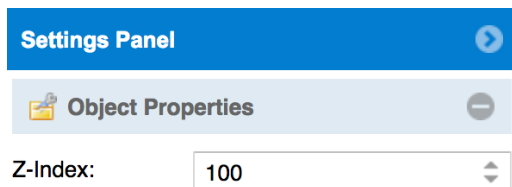
Figure 18. The Z-index

Tab-index

Tab-index is used in order to allow the Test Participant to tab between different input objects such as the input field, radio buttons and checkboxes. This is valuable if the interaction shell contains a form of some sort. This function is presently under development and will only work with a work-around (see below).

All objects that the Test Participant should be able to reach by hitting tab must have a Tab-index. The tab-index must start at "2" in the 2019 version of Ozlab. This is due to that the web browser interprets the address field as tab-index number 1, meaning that if beginning the tab-index settings at 1, the Test Participant's cursor will end up in the address field if tab is hit in the Ozlab Shell. Thus, 2: the first input object reached, 3: the second input object reached, and so forth. Tab-index "2" is shown in Figure 19.

Tab-index is set to objects by clicking either the gear icon behind the object in the Shell objects pane or Scene objects pane or by clicking the object in the Work Area in case of a specific object. The Tab-index can then be set on the right side in "Object Properties" in the Settings Panel.

Figure 19. The Tab-index (marked with red) is found at the bottom of the Properties pane

Hiding objects

Already during shell construction, it can be useful to be able to hide objects from the Test Participant's view. Examples of objects that are normally hidden are error messages and snap points. Items saved as hidden in the Shell Builder will be hidden from the TP's view when the Test Runner is started. The TL may select to sh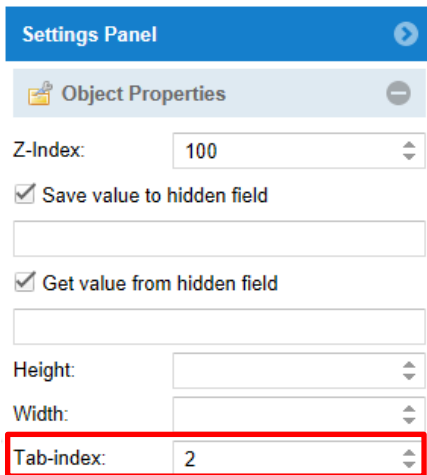ow TP hidden items in the Test Runner when suitable. This is useful, for example, to create the illusion that a field is validated when the TP fills out a form.

⚙ To hide an object, hold the cursor over the objects and then click on the eye in the toolbox (see Figure 20(a)).

Figure 20. (a) Clicking the eye icon hides objects.   **(b) Hidden input fields get a special symbol**

Because "hidden objects" are not hidden to the user of the Shell Builder, and neither to the TL in Test Runner, they need to be marked in some special way. Ozlab lets them be a little bit fainter (see Figure 29 for an example from Test Runner). However, because input fields (whether one-line or multiple-line) consist of just a thin frame, their hidden state is emphasised by a greyed out eye icon with a red bar across it. As shown in Figure 20(b), this symbols is place in the middle of the text field.

Snap points

Snap points can be compared to magnets. Snap points are added to the Work Area. To make an object snap, first select the behaviours MakeObjectSnap and ObjectMobableByTL (and /or Object MovableByTP) in the Object Behaviors panel for that object. The object will be centred over the snap point as soon as the TL (and/or TP) drops the object near the snap point during a session.

⚙ *Adding snap points:* Right-click on the scene. Select "Add snap-point" (see Figure 21). Figure 22 presents a snap point.

Moving snap points: Drag the snap point to the desired position, or use the arrow keys.

Removing snap points: Right-click on the snap point. Select "Remove snap-point".

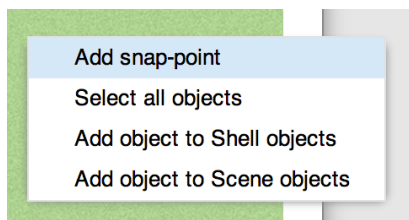❗ Note that some objects might not be movable by TL or TP.
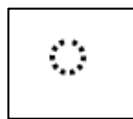
Figure 21. Adding a snap point

Figure 22. A snap point

Enable TP scrolltoTop

If the interaction shell is longer than the screen size, the "Enable TP scrolltoTop" can be used (see Figure 23). If enabled, the interaction shell will jump to the top of the Active Scene when switching between scenes, like most pages on a web site functions.
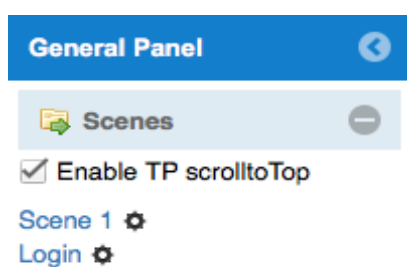
Figure 23. Enable TP scrolltoTop

Alignment line

The alignment line can be used to vertical align objects. Adding an alignment line makes objects "snap" to the line – it is also possible to select several objects at the same time and make them snap to the line. Note that aligned objects do not stay "snapped" to the line for ever. If the alignment line is moved to another location on the scene the objects will not change their placement.

Adding alignment line: Right-click on the scene. Select "Toggle alignment line". The alignment line is added to the scene area.

Moving alignment line: The alignment line can be moved horizontally on the scene area. To move it, click and drag the alignment line to desired position.

Removing alignment line: Right-click on the scene. Select "Toggle alignment line". The alignment line is removed from the scene area.

Making objects uneditable (locking by padlock)

During shell construction, it is sometimes convenient to lock an object so that it is protected by mass-editing of objects when the designer uses ctrl/cmd+A to select all object on the scene. Locked objects is not affected by actions such as drag-and-drop or delete even if all objects on the scene have been selected. To see the status of each object, hover over it and the toolbox in Figure 20(a) will show either an open padlock as in the figure or a closed padlock. Klick on the padlock icon to change its status.

The padlock status is not indicated in the Settings Panel or Object Behaviors.

4. Ozlab on mobile units (including tablets)

4.1.   General

For testing user interfaces on mobile units, most of the Ozlab functions will work as with mouse devices.

To connect the TP mobile unit, start a web browser on that unit (preferably the latest version of *Google Chrome*) and type in the relevant URL. Remember that it is necessary to go via the network where Ozlab is hosted (at Karlstad University it is one of KAU's networks).

In principle, one can use Ozlab also as a TL on a mobile device, but it will be very hard to follow what the TP is doing. Thus, one should only connect to the Ozlab server from a mobile device by using "Join test session" or "View on-going session" (see Figure 24).

⚙ When building interaction shells for mobile phones with the Shell Builder, there are predefined shell sizes. Click "File", then "Set Scene Size…" to select a size.

4.2.   Differences for mobile units

It is possible to use the Vibrate behaviour on mobile phones and tablets, in contrast to cases where the TP is using a computer. On a mobile unit, it is possible to turn the device such that the screen is horizontal (landscape). Ozlab will not adapt to this (as the intention is to design consciously what a scene will look like for the TP), and a blank space will be show to the right of the scene.
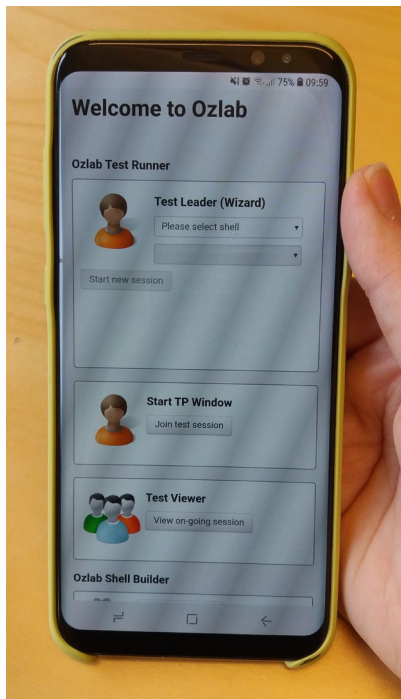
Figure 24. The mobile version of Ozlab's homepage (photo by Elin Nilsson 2018)

Touch input

Touch input includes swipes (see Figure 25) and the possibility to keep several fingers down on the screen simultaneously, but versions of Android older than 6.0 sends at most two touches at the same time (see Figure 26). Click works as usual. TL can only follow TP on the TL screen when TP keeps a finger on the screen. Of course, no mouse pointer will be visible in contrast to computer-based tests where the TL uses a desktop computer or a laptop.

One- and multiple-finger touches are shown in Test Runner mode as one or several balls to the TL. Each ball gets it an individual colour so that the wizard can easily follow the individual touches.

So long as the TP fingers on the screen, a counter will be shown in each ball. Figure 26 shows zero seconds, that is, when the touch just started.

If an object is movable by the TP and it is possible to switch to another scene via swipe, the TP needs to use two fingers to swipe.
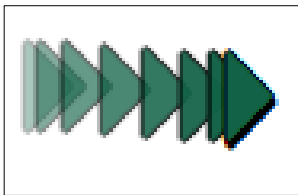
Figure 25. Swipe as seen by the TL
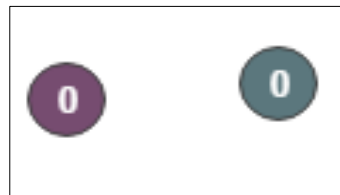
Figure 26. Two-finger touch as seen by TL

5. Test Runner

5.1.   Getting started

Interaction shells created in Shell Builder are run in Test Runner. Test Runner can also be used to practice the interaction shell. The following section introduces how Test Runner works and what parts might be good to know. When Test Runner is running, the interface is green (instead of blue; see Figure 27).
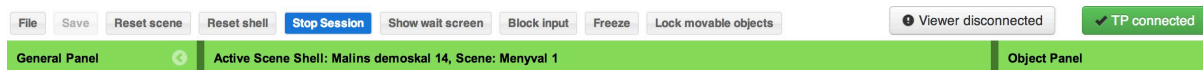
File	Save	Reset scene	Reset shell	**Stop Session**	Show wait screen	Block input	Freeze	Lock movable objects		ⓘ Viewer disconnected	✔ TP connected
General Panel		◀	Active Scene Shell: Malins demoskal 14, Scene: Menyval 1							Object Panel	

Figure 27. Test Runner in Active Session

Start a test session from Shell Builder

⚙ Go to the Work Area of the shell that should appear at the TPs device when the session is started (i.e. the scene that the test is started with). Click "Start Session".

Start a test session from the landing page – TL

⚙ 1.   Start the web browser.
2.   Go to the homepage Ozlab system.
3.   In the column "Ozlab Test Runner", and in the "Test Leader (Wizard)": select the interaction shell to be used in the test in the first drop-down menu "Please select shell".
a. If the test should start in the first scene of the shell, click "Start new session".
b. If the test is to start at any other scene than the first one, select a different scene in the second drop-down menu. Then click "Start new session". Now Ozlab is in the Test Runner mode and a test session is started.

Start a test session – TP

⚙ Go to the home on the same Ozlab site that the TL started the test from. Select "Join test session" under "Start TP Window".

When using Ozlab on distance, distribute the whole TP address including the /tp suffix rather than the web address to the landing page. This will decrease the confusion for participants and there is no risk they start to manipulate any of the shells in Ozlab.

💡 If the TP should not see that there is a browser that the test is run in, the web browser should be switched to full screen mode.

When using mobile devices as TP units, see "Touch input" above about how TP fingers are represented in the TL view of the Test Runner.

Starting a Test Viewer session – TV

⚙ On the observer device, start the web browser, go to the home page of the same Ozlab site that TL started the test from. Select "View on-going session" under "Test Viewer". As for TP, distribute the whole web address including the /tv part to external viewers.

The observer (TV) sees what both the TL and the TP do, but cannot participate in the interaction.

5.2.   Wizard controls

Changing interaction shell during a test session

Click "File" and "Open shell". Then, choose the shell to be opened in the dialog box and click on "Open" (see Figure 28).

For large shells, it can be helpful to split the shell into several shells. Then the lists of scenes and the lists of objects are shorter.
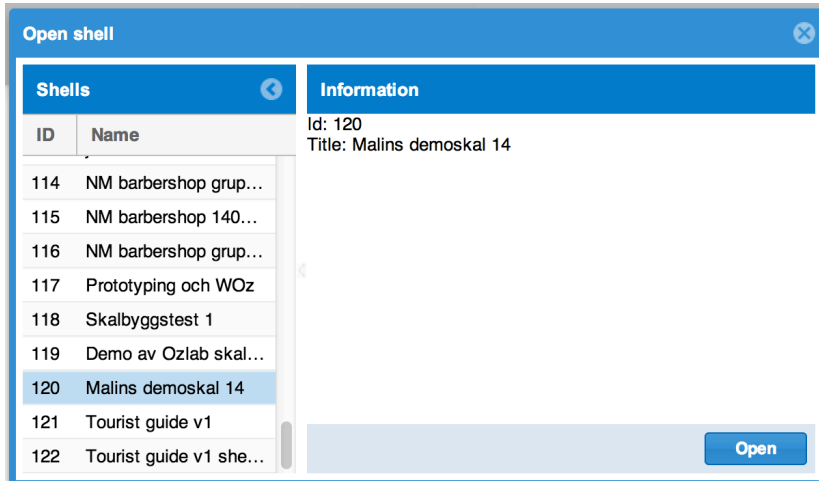

Figure 28. Changing interaction shell

Reset scene

Restores the objects that have been changed during the test at the current scene (e.g. input fields are emptied, draggable items are moved to their original positions, items drawn from Shell objects pane and Scene objects pane are removed from the Active Scene).

Reset shell

Restores the objects that have changed during the test, the entire shell (e.g. input fields are emptied, draggable items are moved to their original positions, items drawn from Shell objects pane and Scene objects pane are removed from the Active Scene, hidden fields are emptied).

Tip: It can be useful to restore the shell/scenes between test subjects.

Stop Session

Exits Test Runner and the test session is terminated. Ozlab returns to Shell Builder.

All changes made in the shell during the test session are lost when the session ends. If something is needed to be changed in the shell (e.g. adding a behaviour to an object), the session can be ended, the change implemented and saved. Then the session can be started again by clicking "Start Session".

Show wait screen

A black screen in the TP' view with the text "Please wait ...". The TP cannot see if the TL makes any changes or switching the scenes behind the waiting screen. View the shell again by clicking "Hide wait screen".

Block input

Displays an "hourglass" (PC) or "beach ball" (Mac) at the TP's mouse cursor, and blocks TP's input (e.g. attempts to write in an input field).

Freeze

Freezes the TP's mouse cursor on the screen.

Lock movable objects

The TP cannot move the object that has the behaviours "ObjectMovableByTP" or "ObjectInvisibleMoveByTP".

The TL can still move objects with the behaviours "ObjectMovableByTL" or "ObjectInvisibleMoveByTL" when the movable objects are locked.

Make TL's mouse cursor visible by TP

Makes the TL's mouse cursor visible on TP's screen.

Changing the Scene size in test session

Scene size can be changed during the test run. Other changes made in Test Runner are not saved. Changing the Scene size during the test session, can be helpful if one is not sure how large the screen of the TP is.

Use shell objects and scene objects

If objects have been added to the Shell objects pane and the Scene objects pane, these objects can be dragged and dropped to the Work Area during a test session.

All objects in the Shell objects pane and Scene objects pane can be dragged to the Work Area even if they lack the behaviours that make them movable. When the objects without these behaviours have been placed on the scene, however, they cannot be moved.

Items that should be centred over snap points on the scene must first be released outside or on the Active Scene.

Show hidden items

Hold the cursor over the object. Shift+click on the object or click on the icon that looks like an eye in the tool box that pops up in the object's right corner as shown in Figure 29. If the object is more transparent than usual, it is hidden from the TP,

Figure 29. The eye icon is used to make hidden objects visible on the TP screen

Move multiple objects simultaneously

Select the objects (use ctrl/cmd and click on each object) and drag one of the objects to move all selected items at once.

6. Practice conducting tests with Ozlab

Before test sessions with real users are made, it can be worthwhile to practice how to control the interaction shell and how the overall test procedures work out.

6.1. Questions for interaction scripts for Wizards

Practice individually or within the design team and ponder question such as:

- What should happen when the test subject does X?
- What should be the actions by the Wizard when the TP does X?
- Which scenes should the Wizard navigate to and when? Should the Wizard use the list of scenes or hidden objects for navigation?
- Are all objects necessary for a test session available in the shell?
- Are all scenes needed within the same shell file, or should the Wizard be prepared to swap shells during a session?
- What tasks is the Test Participant asked to do? These should not be dependent on the fact that one is using an Ozlab shell. (However, rapid prototyping cannot always cover all aspects of an imagined interaction design.)
- Should movable objects be locked during part of a test session?
- When is the test session done? How is this communicated?

6.2. Check the shell

Check by walking through the shell file(s) with a TP browser open that:
- objects have the right behaviours (e.g., movable objects really are movable for TL or TP);
- objects have the right Z-index;
- links to scenes are working.

6.3. Pilot testing is always necessary

Finally, but still before real tests (or demonstrations) start, make at least one test session with someone who has *not* been part of the design work (preferably someone from the target group):

- Does the Test Leader(s) have a clear idea of what to do in different situations?
- Is the introduction to the test (or demonstration) intelligible?
- Are pre-test and post-test questions intelligible?
- If the TP or TV can hear the sound from the TL device, the illusion of a human-to-computer interaction may be ruined – TL must use low volume or headphones.

Record how much time the whole test session takes and its individual parts. This will help planning a real test with several sessions where booking Test Participants in consecutive slots is necessary.

7. Not only tests with Ozlab

7.1. Teaching usability testing by the Wizard-of-Oz method

As Ozlab is web-based it is easy to use in classrooms (Figure 30) and company workshops. Learners can play TL and TP with their laptops, sitting next to each other for better understanding how an interaction shell is built up and what the consequences are for interactive sessions.
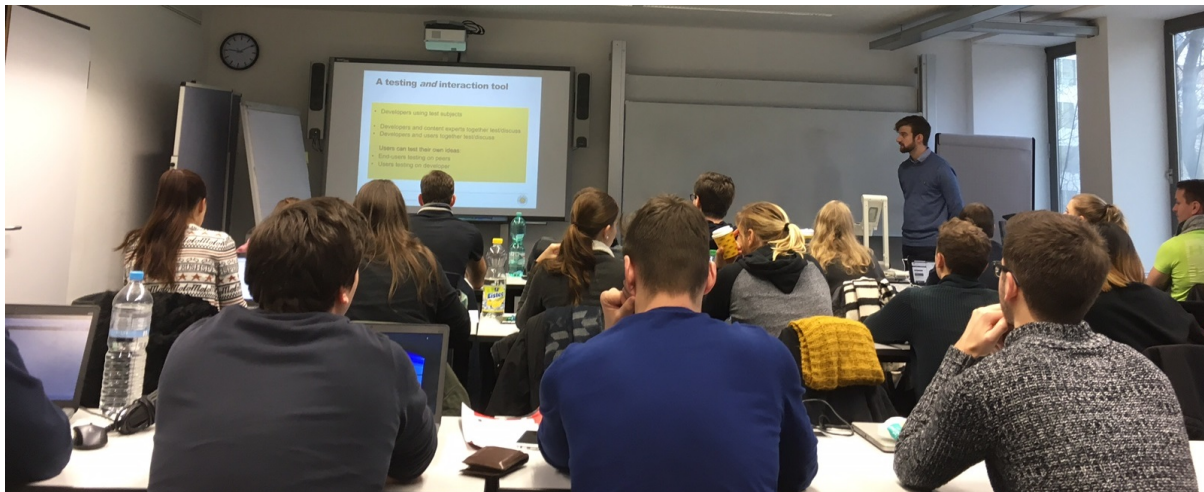

Figure 30 Karlstad University's Henrik Andersson in a demo and workshop session at Duale Hochschule Baden-Württemberg in Stuttgart. (Photo: Kai Holzweissig)

7.2. A testing and interaction tool

The Wizard-of-Oz experimentation does not have to be testing in a proper sense but can be an explorative endeavour, where the wizard is allowed to try out interactive action as his/her understanding increases during an actual interaction with TPs taken from outside the designer team. The Ozlab system is like a sketch book where interaction can be sketched and explored. Thus, there are many uses of a GUI interaction toolbox like Ozlab:

- Professional developers using test subjects

- Developers and content experts together test/discuss
- Developers and users together test/discuss

- Users can test their own ideas:
    - End-users testing on peers
    - (End-)users testing on clients
    - Users testing on developer

The last point may seem puzzling, but developers (programmers) will have questions on how the interaction design will cover certain special cases or special users – the final program will have to be able to handle all sorts of inputs without breakdown. Ozlab lets

them put these questions without using technical language. They will demonstrate the aberrant inputs and TL, here an inexperienced designer, will have to respond GUIly to these in a way TL thinks would benefit the future user (i.e. him/herself and colleagues or friends).

Figure 31, on the other hand, illustrates another kind of expert in the TP role: a content expert. She sits to the right and clicks through WAP pages (not web pages) to see that privacy policy information is given in the right order and of the right format as the designer responds in accordance with his design. Sometimes he interrupts the Test Runner to make some adjustments in wording or placements of text or object, before the session continues.

Figure 31 A session from 2003 where privacy expert TP (right) comments on a flow of WAP pages designed by TL (left). (Photo J.S. Pettersson)

7.3.   GUI-ii: interactive Graphical User Interface interviews

This is the latest methodological development. It exploits the fact that we have a stage to enact GUI interaction on. Furthermore, as we now have a "cloud" solution for the Ozlab functions, we can let people access a shell on an Ozlab web site from anywhere.

- This opens for easy usability testing at distance even if we use the Wizard of Oz method and a highly specialised software

- We can also use it for discussion sessions that only partly takes the form of usability testing

- Thus, we have arrived at a new way to gather user requirements, co-create graphical user interfaces, and evaluate usability – a way that does not presume physical co-location:

the *GUI Interaction Interview* (GUI-ii)

See Figure 32 for a GUI-ii example from project CriseIT (2016-2018) where the TL is speaking and watching, and of course sometimes acting in the GUI. As it is hard to predict the window size of the remote TP, it happens that the scene is somewhat narrower than the TP browser window just as on the laptop in the centre of the figure. This laptop shows the TV view (cf. section 2.1) and it is here used for screen recording. On the main screen, the TL view shows the actual TP window by the thin black frame.

Note that because the scene area is set narrower (albeit higher), TP cannot see the mobile mockup which TL intends to "popup" later in this scene.

The piece of paper under the main screen is TL's personal notes about interaction flow, her "interaction script" (section 6.1).

Sound recording facilities are also illustrated in this figure. The round black Jabra table microphone and loudspeaker is connected to the laptop where Skype provides the voice connection. Alternatively, the ordinary table telephone with hands-free function can be used to sustain the voice communication or a smartphone with good hands-free sound capacity. The screen recorder running on the laptop records both the screen (TV/TP view) as well as the conversation between TP and TL.
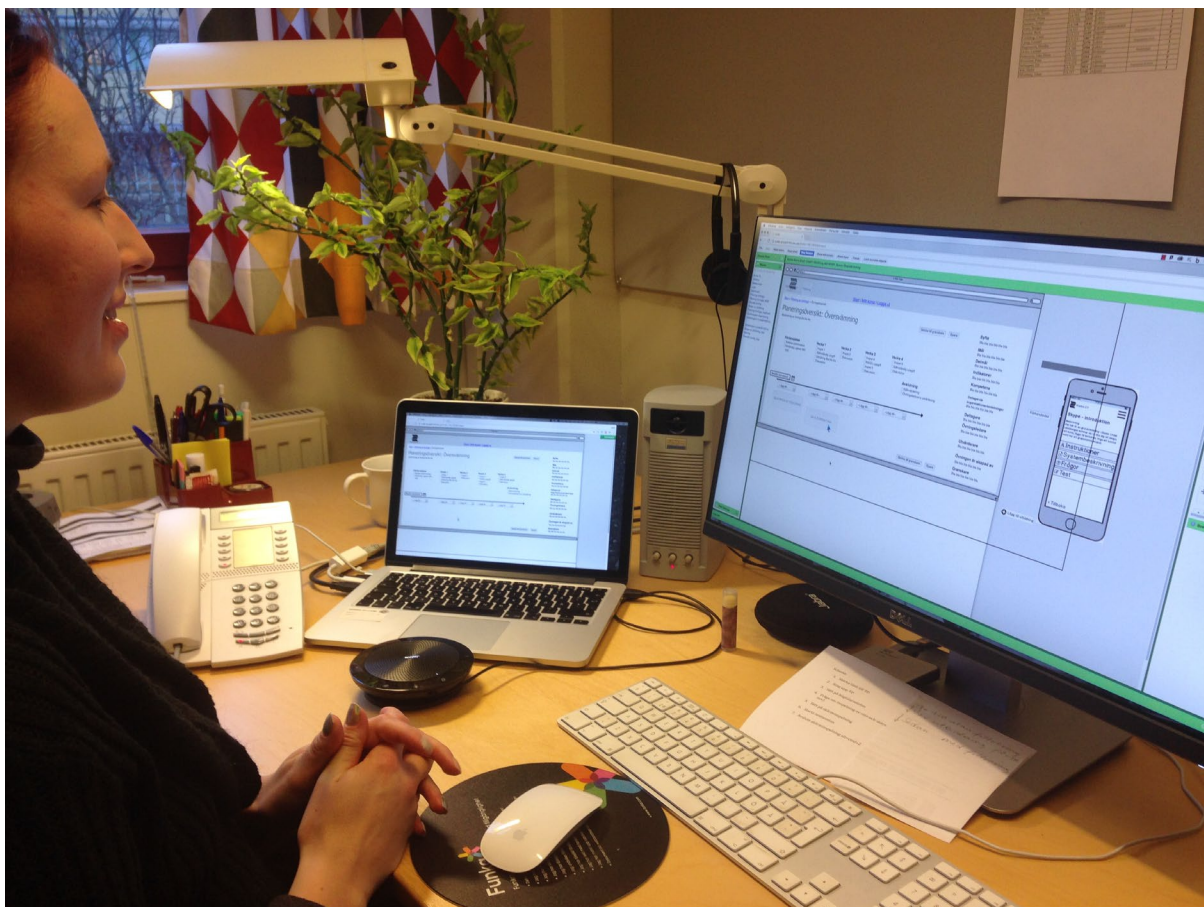

Figure 32 GUI-ii example with a notably relaxed TL (Karlstad University's MalinWik; photo by J.S. Pettersson)