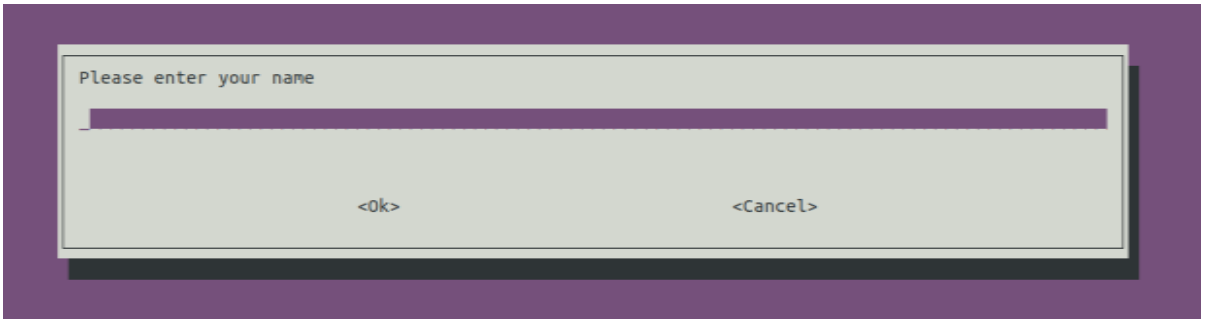
Shell scripts using Whiptail



Have you ever wanted to make your shell script interactive? Or have you ever wanted to display a full screen message from your shell script? **Whiptail** can do this for you!

Whiptail is a program that will let you present a variety of questions or display messages using dialog boxes from a shell script. Currently, these types of dialog boxes are implemented:

It offers a couple of different dialog boxes:

- message box
- yes/no box
- info box
- input box
- password box
- text box
- menu box
- checklist box
- radiolist box
- gauge box

Installation

On Debian based Linux distributions `whiptail` comes preinstalled.

If not you can grab it via:

```
apt-get install whiptail
```

Size of the dialog

All invocations of `whiptail` require you to specify a height (rows) and width (columns) of the dialog box to show. For example `20 100`. It will take some trial and error before the box shows up nicely for your specific content.

Types of dialog boxes

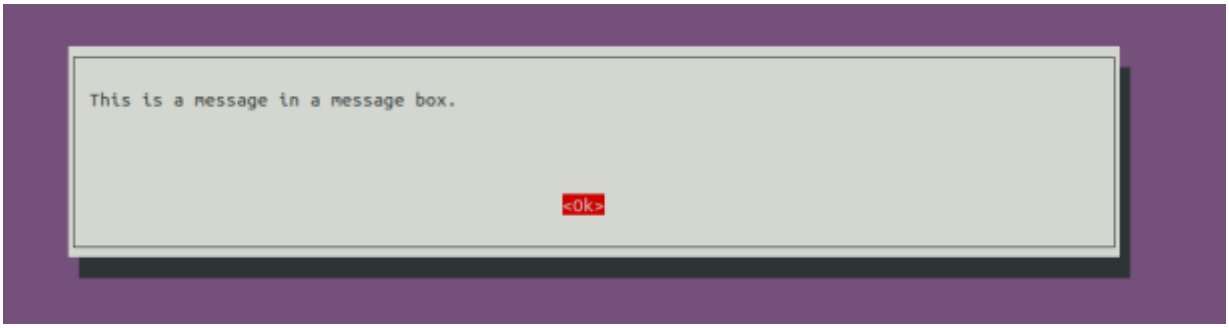
Whiptail offers a couple of different types of dialog boxes.

Message box

A message box can be used to display some text. Script execution is paused until the user hits ENTER.

```
whiptail --title "<message box title>" --msgbox "<text to show>" <height> <width>
```

```
whiptail --msgbox "This is a message in a message box." 10 100
```



Yes/no box

Here you can ask the user a question, by default the `Yes` button is selected

```
whiptail --title "<dialog box title>" --yesno "<text to show>" <height> <width>
```

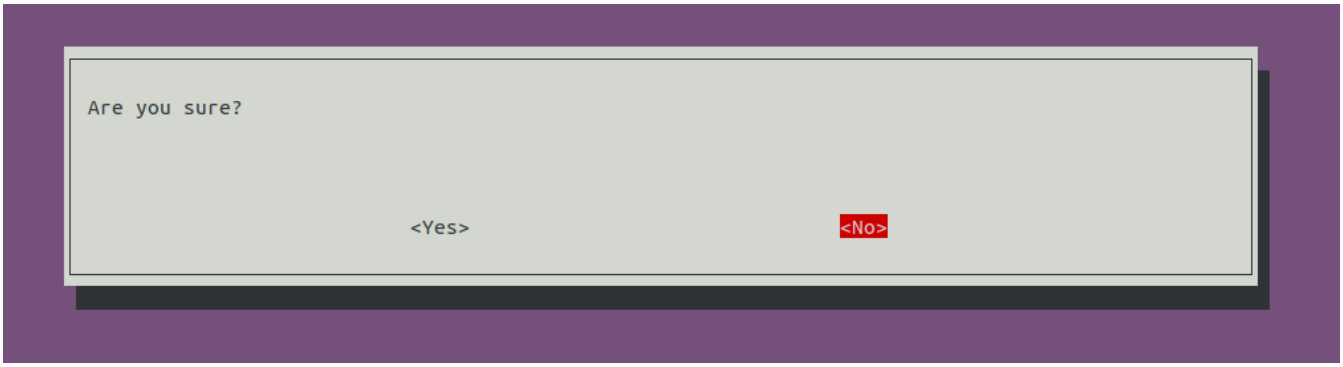
```
#!/bin/bash

if (whiptail --yesno "Are you sure?" 10 100) then
  echo "Yes I am sure!"
else
  echo "No I am not sure!"
fi
```

Use the option `--defaultno` to make `No` the default selected button

```
#!/bin/bash

if (whiptail --yesno --defaultno "Are you sure?" 10 100) then
  echo "Yes I am sure!"
else
  echo "No I am not sure!"
fi
```



Alternatively, you can use the "–yes-button" and "–no-button" options.

```
#!/bin/bash

if (whiptail --title "Test Yes/No Box" --yes-button "Skittles" --no-button
"M&M's"  --yesno "Which do you like better?" 10 60) then

    echo "You chose Skittles."

else

    echo "You chose M&M's."

fi
```

Input box

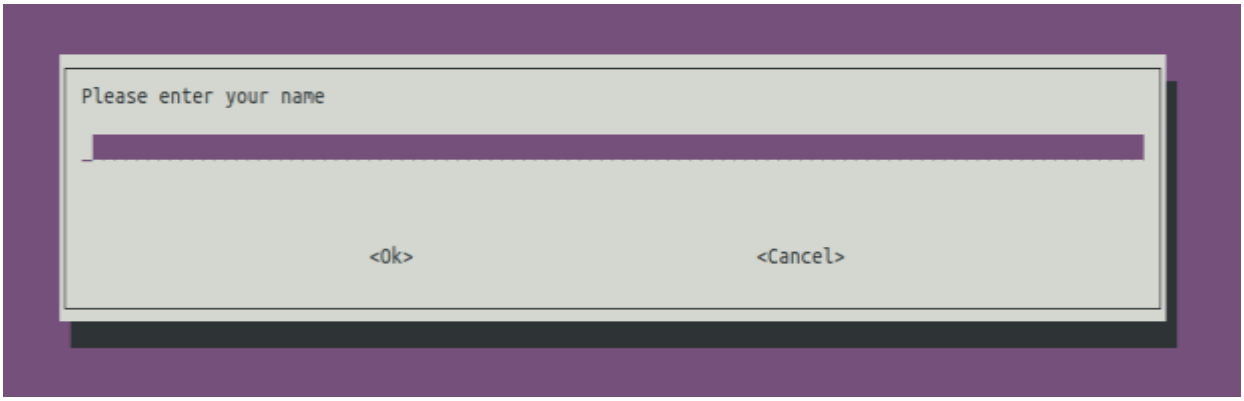
Use an input box to ask the user to answer a question.

```
whiptail --title "<input box title>" --inputbox "<text to show>" <height> <width>
<default-text>
```

```
#!/bin/bash

NAME=$(whiptail --inputbox "Please enter your name" 10 100 3>&1 1>&2 2>&3)
echo "name: $NAME"
```

The `3>&1 1>&2 2>&3` part switches the `stdout` and `stderr` file descriptors. This is needed because we want to assign the user input to the variable `NAME`. This variable will be assigned whatever the subshell command outputs to `stdout`, only problem is that `whiptail` prints the input string to `stderr` by default.



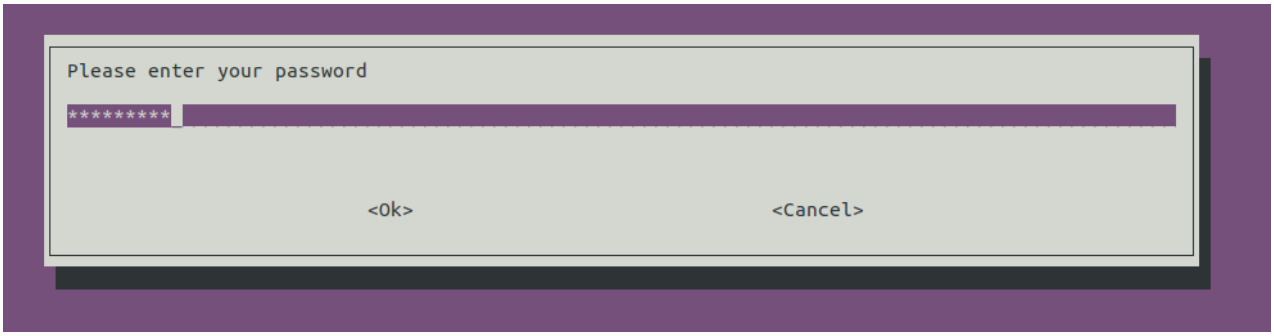
Password box

A password is just like an input box however it displays an `*` for every inputted character.

```
#!/bin/bash

PASSWORD=$(whiptail --passwordbox "Please enter your password" 10 100 3>&1 1>&2 2>&3)

echo "name: $PASSWORD "
```

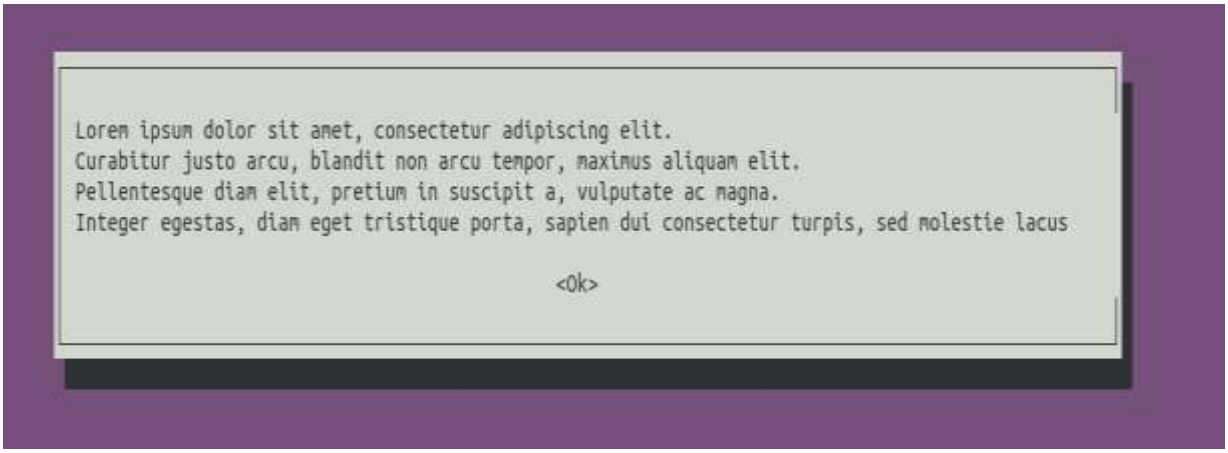


Text box

A text box can be used to display the contents of a file. The `--scrolltext` option makes sure the user can scroll the dialog box.

```
#!/bin/bash

whiptail --textbox file.txt 10 100 --scrolltext
```



Menu box

A menu can be used to present multiple options and having the user choose one option. Every menu item consists of a tag string and an item string. The tag string is the name of the menu item - and will be printed to `stderr` (we use the stderr/stdout flip trick again to print the selected option to `stdout` instead) when the user hits ENTER. The item string is a description of the menu item. Whiptail does not enforce tag strings to be unique.

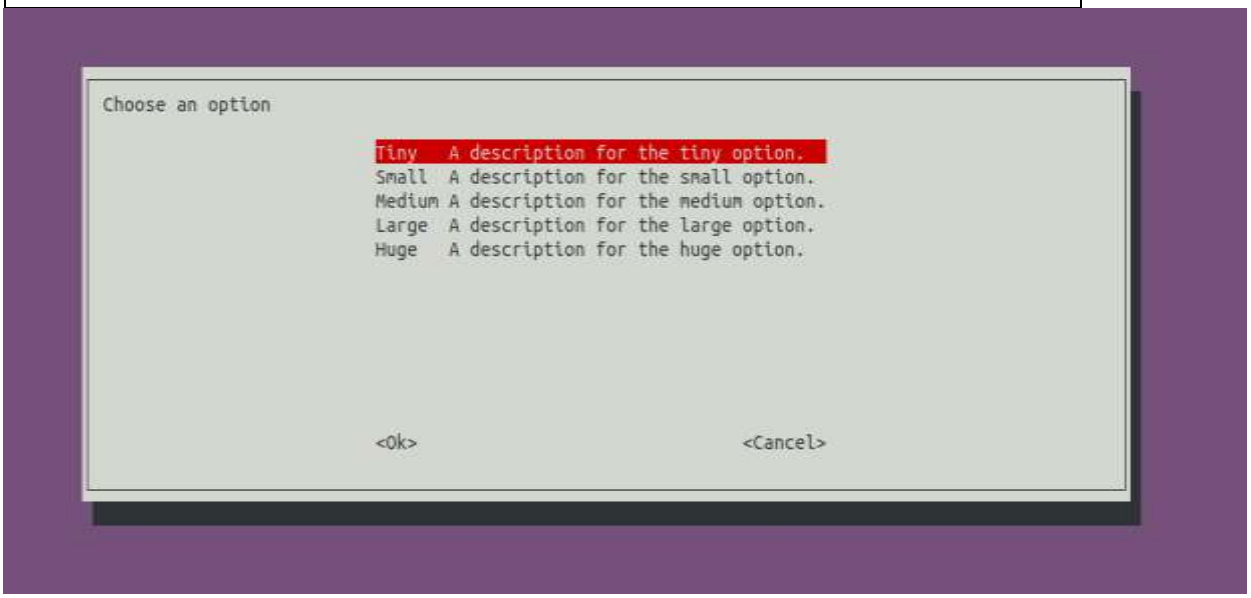
After the height and width of the dialog box, another parameter is required: the height of the menu list. You probably want this value to be lower than the height of the dialog box.

```
#!/bin/bash

CHOICE=$(whiptail --menu "Choose an option" 18 100 10 \
  "Tiny" "A description for the tiny option." \
  "Small" "A description for the small option." \
  "Medium" "A description for the medium option." \
  "Large" "A description for the large option." \
  "Huge" "A description for the huge option." 3>&1 1>&2 2>&3)

if [ -z "$CHOICE" ]
then
  echo "No option was chosen (user hit Cancel)"
```

```
else
  echo "The user chose $CHOICE"
fi
```



Checklist box

A checklist box is similar to a menu box, but it allows the user to select zero or more options.

The list dialog is useful when you want the user to select multiple options in a list. The radiolist dialog only allows one to be selected. Syntax:

```
whiptail --title "<checklist title>" --checklist "<text to show>" <height>
<width> <list height>
```

Examples:

```
#!/bin/bash
CHOICE=$(whiptail --title "Test Checklist Dialog" --checklist \
"Choose preferred Linux distros" 15 60 4 \
"debian" "Venerable Debian" ON \
"ubuntu" "Popular Ubuntu" OFF \
"centos" "Stable CentOS" ON \
"mint" "Rising Star Mint" OFF 3>&1 1>&2 2>&3)

if [ -z "$CHOICE" ]
then
  echo "No option was chosen (user hit Cancel)"
else
  echo "The user chose $CHOICE"
fi
```

Radiolist box

A radiolist is similar to a menu box but it allows you to set a default selected option.

```
#!/bin/bash
CHOICE=$(whiptail --title "Test Checklist Dialog" --radiolist \
"What is the Linux distro of your choice?" 15 60 4 \
"debian" "Venerable Debian" ON \
"ubuntu" "Popular Ubuntu" OFF \
"centos" "Stable CentOS" OFF \
"mint" "Rising Star Mint" OFF 3>&1 1>&2 2>&3)

if [ -z "$CHOICE" ]
then
  echo "No option was chosen (user hit Cancel)"
else
  echo "The user chose $CHOICE"
fi
```

Gauge box

The last type, the gauge box, can be used to display a progress bar. Whiptail reads new percentages to update the progress bar from `stdin`.

Create a progress bar

The progress bar is a user-friendly dialog. Whiptail reads a percentage (0 to 100) from standard input and displays the corresponding count in a table.

grammar:

```
whiptail --gauge "<test to show>" <height> <width> <inital percent>
```

Example:

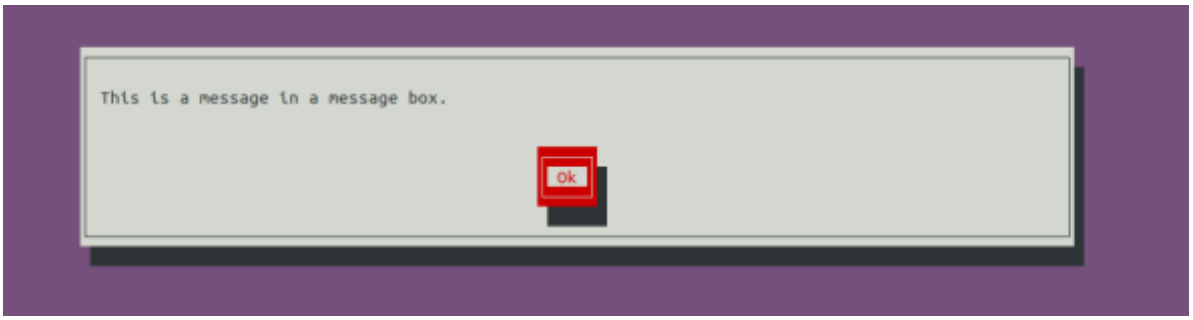
```
#!/bin/bash
{
    for i in {0..100..10}
     do
        sleep 1
        echo $i
    done
} | whiptail --gauge "Please wait while installing" 6 60 0
```

Other Options

Some options you may find useful:

```
--fb
```

Display Full Buttons with a 3D-effect.




```
--nocancel
```

Do not show a Cancel button. Can for example be used together with a menu box.