Build and Share Interactive Content with Maple 2025

Maple 2025 provides options for creating documents and interactive applications programmatically. Maple can be used to develop content that can be used interactively within Maple, or automatically translated for use within Maple Learn. Maple Learn is an online environment designed specifically for teaching and learning mathematics.

Sections

A new Section element has been added to the Canvas package. Group content can now contain collapsible sections. These are helpful for organizing content, and can also be used to hiding content that you don't initially want students to see. In this example the Solution section is initially collapsed.

```
with (DocumentTools  :-Canvas ):
cv  :=  NewCanvas  ( [
     Text ("Sketch the graph of: %1"   ,3^(1+2*x)),
     Group ([
         Section ([
             "Solution" ,
             StaticPlot (plot (3^(1+2*x)))
         ],'expanded' =false )
     ])
]):
ShowCanvas (cv,entrybox =  false );
```

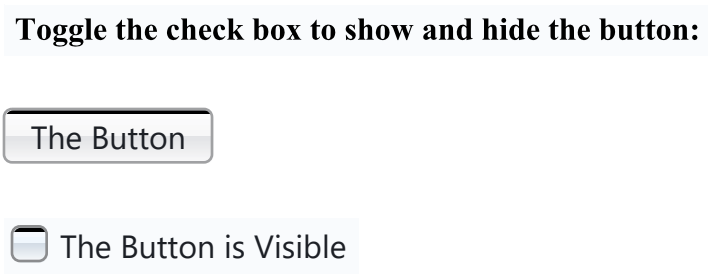
Sketch the graph of: 3^{1+2x}

▶ Solution

Check-Box Actions

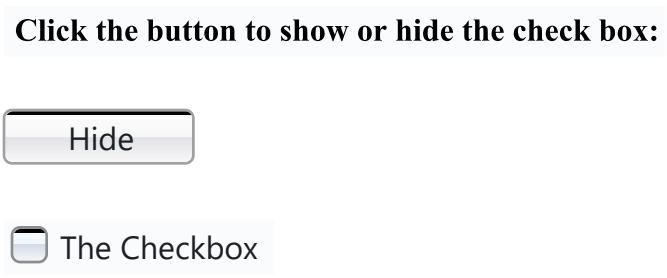
The ScriptCheckBox command is used to insert a check box into an interactive application. It can now be given an action that is invoked when the check box is selected or cleared. In this example, the check box has an action handler to show and hide the button immediately when you click it.

```
with (DocumentTools :-Canvas ):
showhide := proc ( canvas )
    local M, sc;
    M := GetElements (canvas ,"checkbox" );
    sc := Script ();
    SetButton (sc,"The Button" ,"visible" ,M[1]:-ischecked );
    ToString (sc);
end proc :

cv := NewCanvas ([Text ("Toggle the check box to show and hide the button:"     ,bold ),
                  ScriptButton ("The Button" , ""),
                  ScriptCheckBox ("The Button is Visible"  , showhide , 'ischecked'  = true , 'custom'  = "cb")
                  ]):
ShowCanvas (cv,entrybox = false );
```



Additionally, check boxes now have a 'visible' property so they can be shown or hidden as the result of any action.

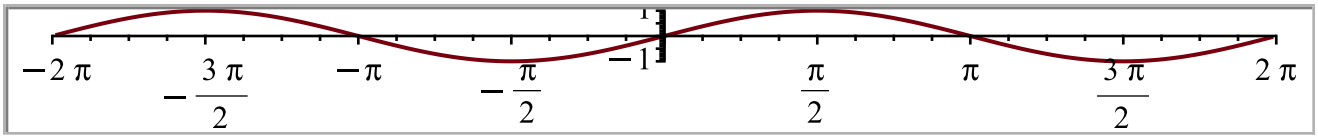
```
with (DocumentTools :-Canvas ):
showhide := proc ( canvas )
    local cb, sc;
    cb := GetElements (canvas ,"checkbox" )[1];
    sc := Script ();
    if cb:-visible = true then
         SetCheckBox (sc,"cb" ,"visible" ,false );
         SetButton (sc,"Hide" ,"caption" ,"Show" );
    else
         SetCheckBox (sc,"cb" ,"visible" ,true );
         SetButton (sc,"Show" ,"caption" ,"Hide" );
    end if;

    ToString (sc);
end proc :

cv := NewCanvas ([Text ("Click the button to show or hide the check box:"     ,bold ),
                  ScriptButton ("Hide" , showhide ),
                  ScriptCheckBox ("The Checkbox" , 'ischecked' = true , 'visible' = true , 'custom' = "cb" )
                  ]):
ShowCanvas (cv,entrybox = false );
```

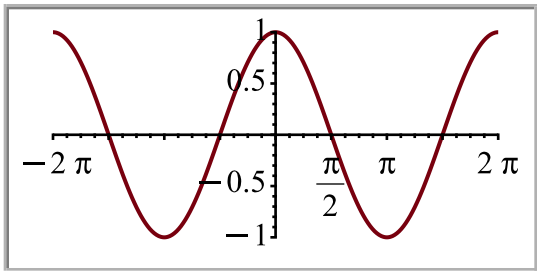


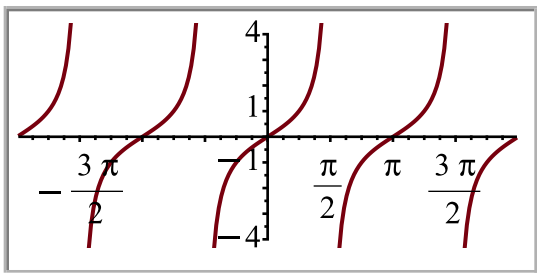
Plot Size

Canvas-based plots now accept width and height parameters.

```
with(DocumentTools:-Canvas):
cv := NewCanvas( [ StaticPlot( plot(sin), width = 1000, height = 100 ),
                   StaticPlot( plot(cos), width = 200, height = 100 ),
                   StaticPlot( plot(tan), width = 200, height = 100 )
                 ] ):
ShowCanvas(cv, entrybox=false);
```





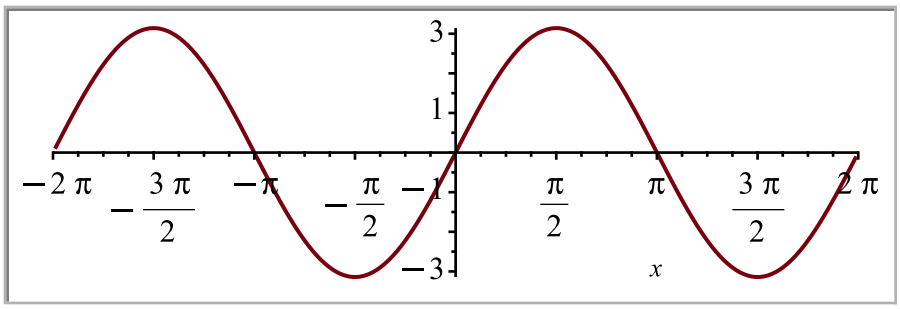


Similarly, the Quiz command respects the 'plotsize' option when outputting canvas-based quizzes.

```
Grading:-Quiz("Which is a plot of ${FUNC}?"   , 1,
    proc() local A;
       randomize();
       A := [sin(x), cos(x), tan(x)][rand(1 .. 3)()];
       Grading:-Quiz:-Set(`$FUNC`  = A);
       [plot(A), plot(-A), plot(Pi*A,x = -2*Pi .. 2*Pi)];
    end proc,
    gradeisindex  = true,
    title  = "Plot Multiple-Choice Randomized"   ,
    style  = multiplechoice ,
    inertform  = false,
    output  = showcanvas ,
    plotsize=[600,200]
);
```

Plot Multiple-Choice Randomized

Which is a plot of $sin(x)$?



Check Answer

Try Another