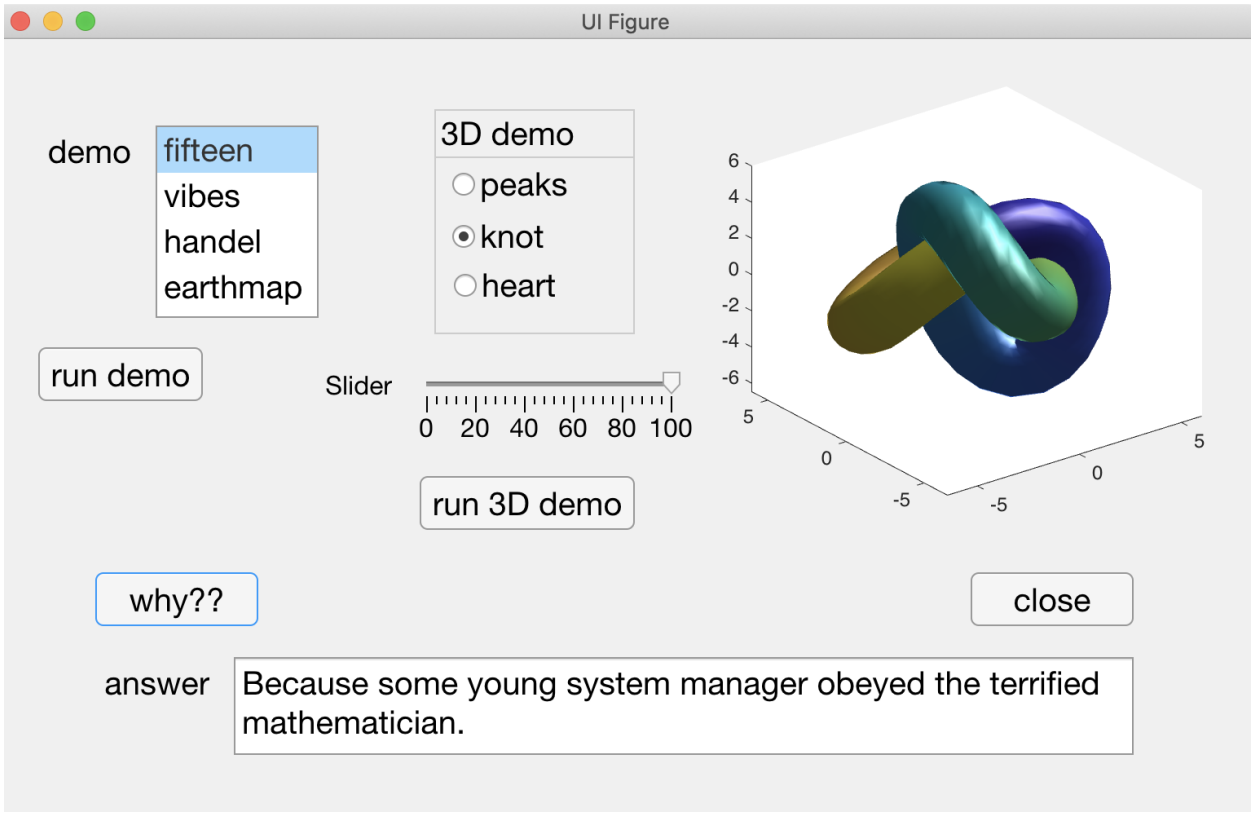
Video #6: Having Fun with a MATLAB App

In this final App, we're going to have some fun with things you can do in MATLAB, and learn about a few extra GUI components along the way. My App is called funApp, and let's first explore. We have a list of demos on the left that are all built-in demos in MATLAB. We can make a selection and then press the run demo button - the first allows us to play the 15-tiles-puzzle game (we'll leave that for later). Then vibes, which we can speed up or slow down. Listen to some Handel. Finally, we can view the world with the earthmap demo. We have some more 3D demos in the middle that we can select with the radio buttons and run with the "run 3D demo" button. The first two also use the slider. Let's first look at the peaks demo - we can adjust the slider (100) and run the demo again and it stretches out the function in the vertical direction. Here we have a chubby knot, can move our slider down and make a skinny knot. The heart doesn't use the slider, and takes more time to compute, but there, a 3D heart. The final demo is based on the built-in why function in MATLAB - generates a random response to the question, why?



So what can we learn from this new App, other than someone at Mathworks has a sense of humor. Let's view the App in App Designer and first have a look at the Design View. There are four new GUI components here that we haven't seen in the previous videos. On the left we have a List GUI component. A List is just like a Drop Down menu, except that all the items in the list are visible at the same time. Its properties are similar to the Drop Down menu - there's a Label (demo), Items, and Value, which is the item in the list that the user has selected. To create the radio buttons, I first dragged a Radio Button Group to the canvas - here MATLAB creates a name for the group, which I changed to app.demoButtonGroup, and there are individual names for each button. The ButtonGroup has a Title, I set to 3D demo, and then we can click on each individual button to set

the text we want to appear next to the button, and each button has a Value property that indicates whether it's selected. Only one button in the group can be selected at one time. Even at the outset, if I decide I want the heartButton selected, the peaksButton will automatically be unselected.

The next new component is the Slider. It also has a Label (I just left it as Slider), and you can specify the range of values on the slider to be whatever you want - if you click on this box on the right with the three dots, it opens up a box where you can specify a minimum and maximum value for the range displayed on the slider, e.g. 10-20 (display adjusts), and note that a slider can also be horizontal or vertical <demo>. The value that the user set within its range is stored in the Value property for the slider, and we can set that whatever we want, e.g. if I want the slider to be set in the middle of its range at the outset, I can change the Value to 15.

The last thing here is this text box at the bottom that's referred to as a Text Area. It's a place where you can place multiple lines of text. This text is stored in the Value property for the Text Area, and you can have text there at the outset. By default, the user can edit the text in this box, but you can make it not editable by the user by unchecking the Editable property in the Interactivity area.

We'll take a peak at the code in the Code View - it's all pretty straightforward. When we push the why?? button, a random string of text is generated by the why function in MATLAB and stored in the Value property for the Text Area that we just learned about, which displays the text on the GUI. When we click on the close button, the app is deleted to terminate the program.

```
% Button pushed function: whyButton
function whyButtonPushed(app, event)
    % modified from MATLAB's why function
    app.answerTextArea.Value = why;
end

% Button pushed function: closeButton
function closeButtonPushed(app, event)
    delete(app)     % terminate the program
end
```

When we push the run demo button on the left, this rundemoButtonPushed function is called. It gets the user's selection from the List GUI component, and depending on the string they selected, a different demo is run (15-tiles puzzle, vibes, play a Handel clip, or show the earthmap). And finally, when we push the button that says "run 3D demo", this function run3DdemoButtonPushed is called, and here, we check the Value of the individual radio buttons to see which one is selected. If the peaksButton is selected, we run code for the peaks demo, otherwise if they chose the knotButton, we display the knot, otherwise the heart.

```
% Button pushed function: rundemoButton
function rundemoButtonPushed(app, event)
    demo = app.demoListBox.Value;
    if strcmp(demo,'fifteen')
        fifteen            % play the 15-tiles game
    elseif strcmp(demo,'vibes')
        vibes              % see cool vibes
    elseif strcmp(demo,'handel')
        load handel        % listen to Handel
        sound(y,Fs)
    else
        figure
        earthmap           % see the world
    end
end
```

```
% Button pushed function: run3DdemoButton
function run3DdemoButtonPushed(app, event)
    if app.peaksButton.Value
        % use slider to show a scaled peaks function
        scale = app.Slider.Value;
        cla(app.plotAxes)
        surf(app.plotAxes, scale*peaks)
        view(app.plotAxes,0,0)
        axis(app.plotAxes, [0 50 0 50 -1000 1000]);
    elseif app.knotButton.Value
        % modified from MATLAB's knot function
        knot(app.plotAxes,app.Slider.Value)
    else
        % show a 3D red heart
        showHeart(app.plotAxes)
    end
```

So that's it for our last example of building an App with a graphical user interface, using MATLAB's App Designer. You should now be set with all the tools you need to build your own Apps in the new assignment.