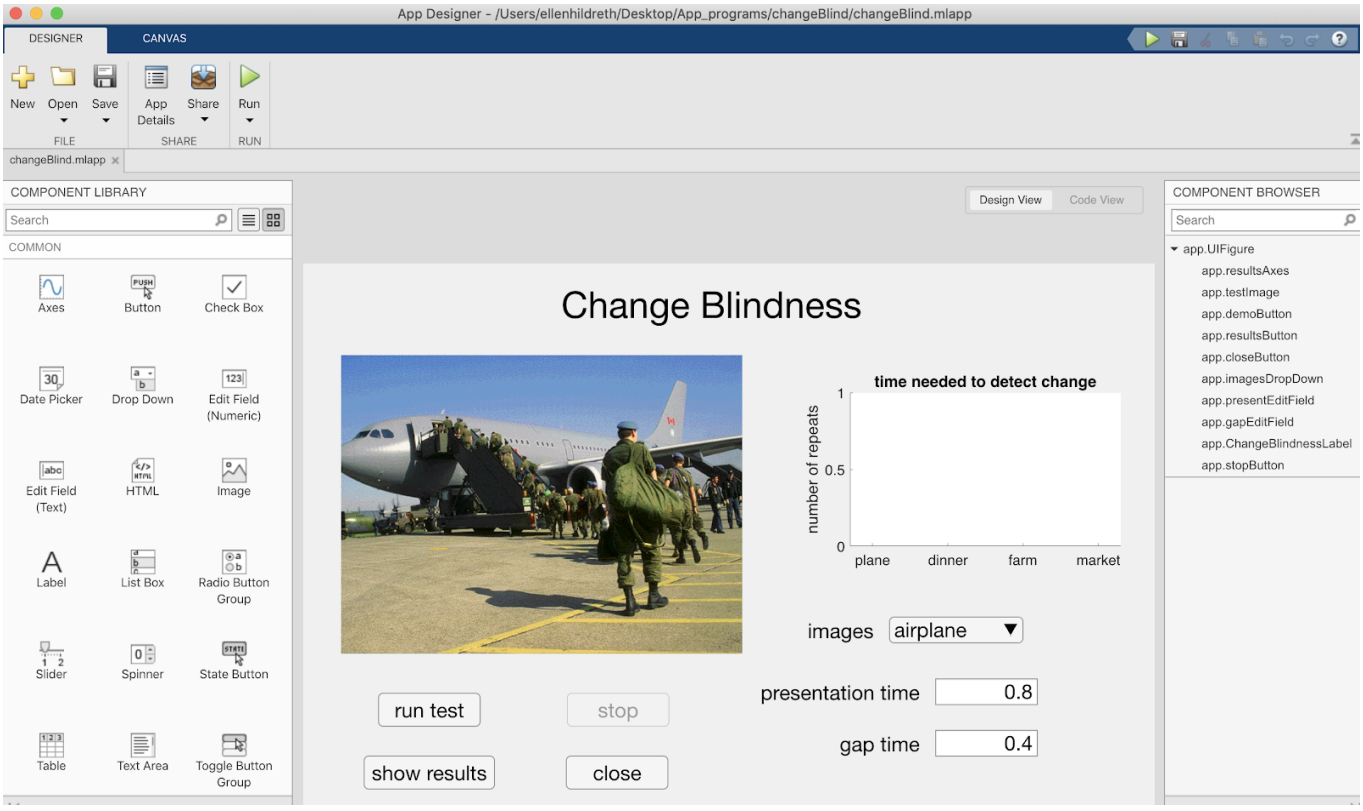
Video #4: An App to Explore the Limits of Visual Perception

In the next two videos we'll explore an App that demonstrates a phenomenon in visual perception known as change blindness - our remarkable ***inability*** to notice large changes taking place in our visual environment. You can see some nice videos of this phenomenon on YouTube if you search for "change blindness." I'll demonstrate the phenomenon here, with an App that I created. When I click on the "run test" button, you'll see a movie on the left that alternates between two images of this airplane scene, and something changes between the two images. Your task is to notice what changed. Let's start the movie. <start demo> I'm going to pretend you see the change and stop the movie. In case you didn't see the change, don't despair, most people don't see it in such a short time. I'll run the movie again, but this time give you a big hint - pay attention to the area around the engine underneath the wing of the airplane. <restart demo> Hopefully you see the change now, and I'll again stop the movie.

In the Drop Down menu here, there are three other choices for the images, and I'll run through each one. I may not give you enough time to notice what changed, but you can just replay the video. It will be tempting to just focus on the movie, but also notice what happens with this stop button as I run the demo - right now I can't interact with this button, it's disabled, but once I start running the movie, it will become enabled so I can click on the button to stop the movie. <demo other images - dinner, farm, market>

As I ran each movie, the program kept track of how many times it cycled through the pair of images before the stop button was pushed to indicate that the change was spotted. Now that I've run through all the image pairs, I'll show the results. We see a bar plot that shows the number of times the images were repeated before seeing the change, for each of the four scenes.

You may wonder what the presentation and gap times are. When I show the movie, each image stays on the screen for a certain length of time, the presentation time - this is seconds, so it's 0.8 secs and we can change this (0.5). There's also a gap in time between the images that we can change. Suppose we take it away (0.0) <run test> - now there's a sudden change between the two images that we can detect more easily, so having a gap in time between the two images is critical to the phenomenon of change blindness. In the rest of this video, I'll just highlight some new things about the visual layout created in the Design View of App Designer. In the next video, we'll look at the code that was added to implement the actions in this program. I'll close the program and switch to my App Designer window.



There are several new things about this visual layout that I'd like to highlight. The images here are displayed on an Image component that you can see in the palette. You can also display images in an Axes area, but the Image component is made specially for this purpose. We can see the properties of an Image component in the Inspector on the right, and notice the ImageSource property that stores the initial image that appears when we start the App. The Browse icon (circled in red below) allows us to navigate to an image file to use for this. We've used the Axes component before, but here I'd just like to highlight some of its properties - the strings that appear as the Y label and title are set up in advance, along with the XTicks and XTickLabels that are the words on the X axis (circled in blue below). We had a Drop Down menu in our energyApp program, and we also had Edit Fields, but here, we want the user to enter numerical information, so I dragged Edit Field (Numeric) components to the canvas in this case.

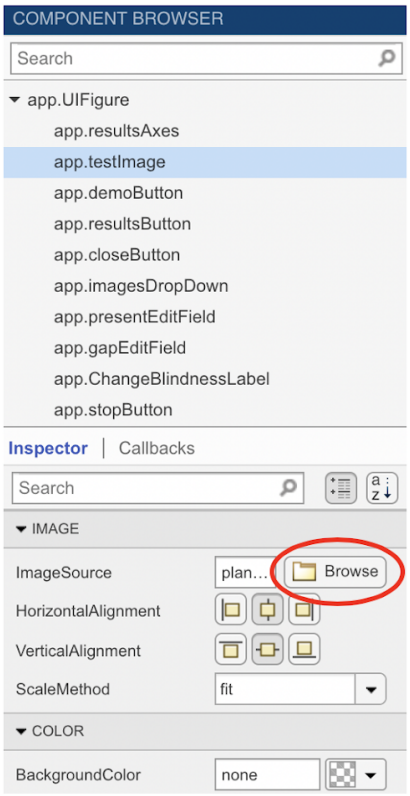
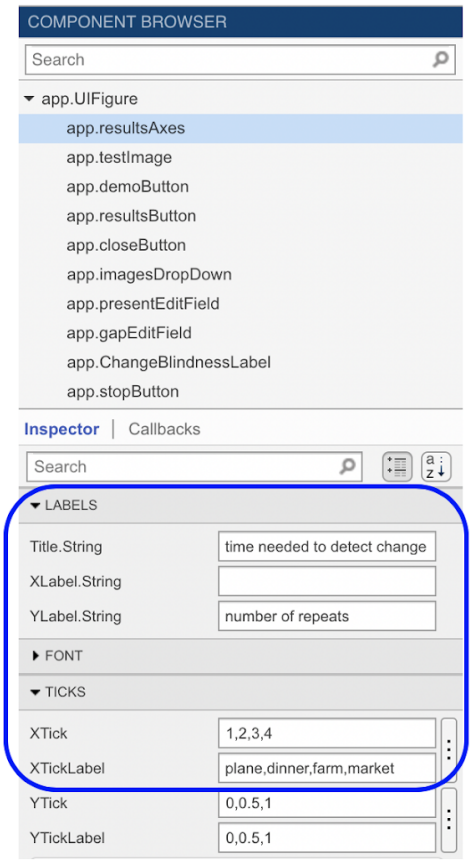
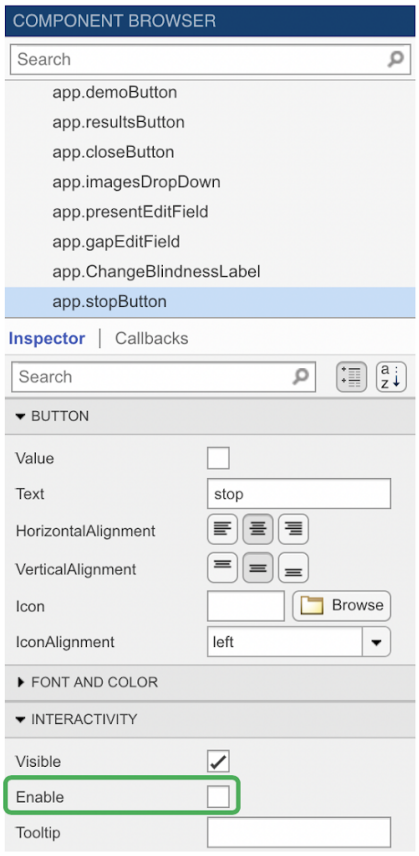


Image component



Axes component



All components

Three of the four buttons are simple buttons, but the stop button is a State Button - we use this button to stop the movie. It begins in the off state with its Value property unchecked, and clicking the button while the movie is running will change it to an on state, and stop the movie. I noted earlier that this button is disabled at the outset, preventing the user from being able to interact with it when we start the program. If we look at the Interactivity properties in the Inspector, we see a property named Enable. Every GUI component has this property, and by default, it's checked, allowing the user to interact with that component from the start. But in this case, we want to disable the button at the outset, which means turning the Enable property to its off state, unchecking this box. That's an overview of the visual layout of the graphical user interface for this App. In the next video, we'll look at how we code the actions for this program.