Hand design exercise

Background:

You'll be working with a virtual prosthesis recreated based on Jumpei Kashiwakura's design. You'll start with a half-complete MJCF file, and you'll complete it and extend it with additional functionality.

Instructions:

Open the `hand.xml` file in a text editor of your choice. Examine the structure of the MJCF, you should see the familiar hierarchical body tree structure. A big difference from other models you've seen during this workshop is the inclusion of mesh assets, they are declared within the `<asset/>` element, and referenced in their corresponding mesh type `geoms`. You'll notice there's only one finger with joints, and it doesn't behave as you'd expect the real device to. You'll need to add missing degrees of freedom, edit joint parameters and add actuators to get the full hand functionality. To test your arm, simulate the `hand_scene.xml` file, which adds a ground plane and lighting to your model.

To make your life easier, try to declare default values for elements that will be reused for all element of that type (documentation).

Goals:

- Be able to control all fingers of the hand with actuators and control the thumb deviation and flexion separately.
- Adjust the parameters of the joints so they behave more realistically. Set them realistic range of motion (see the flexion joint for an example in the file), and experiment with the `stiffness`, `damping` and `springref` (passive equilibrium position) fields.
- Familiarise yourself with the MuJoCo documentation to answer questions you might have about available elements for the MJCF. The documentation is available at this link:
  https://mujoco.readthedocs.io/en/stable/XMLreference.html

Bonus goals:

- Rotate the base body of the hand with 90 degrees using the `euler` field to have the arm stick out over the surface, and move it up using the `pos` field, so it does not intersect with the ground plane. Add an object to your scene within reach of your hand and try to pick it up!

- Contacts are notoriously tricky to resolve with complex meshes. You can get more stable interactions if you disable collisions (see `contype` and `conaffinity` fields) for the mesh geoms and add surrogate colliders based on primitive shapes (e.g., cylinders and boxes) over the fingers and palms. You can set the `group` field of these geoms to 4 to be not visible by default, and then you can enable the group to see them in the bottom dropdown of the left sidebar.
- You can add sensors to simulate data acquisition from the arm. Try experimenting with `touch` type sensors (documentation) to add surface force sensors to the fingers (you can display sensor data with F4).
- Add a `camera` element to your model attached to the palm to simulate an embedded vision system in the hand. You can cycle what camera is active with the "[" and "]" keys.
- Use a `mocap` body and a `weld` equality (documentation) to set up a moveable arm, the position of which you can interactively control with your mouse.
- Attach your hand to a prosthetic arm! Examine the compose_model.xml file in the "elbow" folder next to the arm design one. It uses a package from the DeepMind Control Suite to combine the model file of a prosthetic arm with a target hand. This way you can programmatically compose scenes instead of manually needing to make changes.