Tutorial

# Chrome Developer Tools

## HTML and CSS Inspection

## JavaScript Debugging

## Device Mode

## Remote Debugging

# XD Testing Tools

## Emulating Devices

**Adding a new device:** Click the + button in the menu on top. You can either select a predefined device by starting to type the name of a device in the “Predefined devices” input field or by typing in the specifications of the device in the other input fields.

**Moving the device:** You can drag and drop the device around the border or title of the device and move it to the desired position.

**Settings of the device:** By clicking on the settings button on the top right of the emulated device, you can open a settings panel. You can refresh a device from there, scale it down or up (won’t change the actual resolution of the device). You can also connect the device to another device by selecting the other device from the dropdown menu.

## Connecting Devices

**Adding real devices:** To add a new device, you can click on the QR code button on the top right and scan the QR code, or just type the URL <http://129.132.173.2/remote.html> into the Chrome browser of the device.

**Auto-connecting:** In the sessions area of the tool, you can check the auto-connect checkbox of a device. All newly created actual and emulated devices will automatically connect to that device.

**Manually connecting:** Open the settings menu of the device and choose the device you want to connect to from the dropdown list.

**Resetting a session:** If you click on the “Reset session” button in the session area, all devices will be assigned a new ID and will be reconnected.

## Record/Replay

**Recording interactions:** On the right side of the tools, you see the record/replay area. Click on the record button of a specific device, then perform the desired interactions and click the same button again.

**Saving event sequences:** You can save an event sequence by clicking on the save button at the top right of the event sequence, then typing a name and hitting enter.

**Appending event sequences:** To add a saved event sequence to a device, select it from the dropdown menu at the top of the device’s section in the record/replay area.

**Adjusting event sequences:** Once you have recorded an event sequence, you can drag and drop it to a later point in the timeline or to another device.

**Adding breakpoints:** To add a breakpoint, click at the desired location of the breakpoint to the left of the timeline labels between the two thin vertical lines. To remove the breakpoint again, start dragging it and drop it into the icon that will be highlighted on top. When a breakpoint is reached during replay, the replaying of interactions will pause until you click on the play button on top of the breakpoints.

**Replaying event sequences:** To replay an event sequence on a specific device, click that device’s replay button. To replay the sequences on all devices, click the play button on top of the page.

## Shared JavaScript Console

**Activating/deactivating devices:** Above the console, you see the names of all devices. By clicking on a device you can activate or deactivate it. Each active device has its own unique color which is used to color-code the console outputs from the devices.

**Sending commands:** If you type a JavaScript command into the input field at the bottom of the console and hit enter, the JavaScript command will be executed on all active devices.

**Aggregating console outputs:** The console aggregates the console outputs from all active devices. This includes functions like console.log, console.warn, … It also displays the return values of all JavaScript commands that were sent to the devices. It also forwards all JavaScript errors that occur during execution.

## Function Debugging

**Starting function debugging:** After adding all emulated devices that you want to debug, open the Chrome Developer tools (if they are already open, close them and re-open them). Next to the JavaScript console, a new area will appear for function debugging.

**Debug functions:** To debug a function, add it to the debug list by typing its name in the input field on top of the “Function debugging” area. If the function is then called from any active device, execution will be paused, similar to setting a breakpoint in the first line of the function. You can also look at a function by clicking on the inspect button next to its name.

## Shared CSS Editor

TODO