Locator Creator – Chrome Extension

A Chrome Extension that allows QA engineers and automation developers to **hover, click any element on a webpage, and instantly generate Playwright and Selenium locators**.

The extension provides:

- Visual element highlighting
- Smart locator generation (ID, data-testid, class, XPath)
- Validation for uniqueness
- Checkbox-based UI to choose which locator(s) to copy
- Clipboard integration

---

1. Project Structure

```
locator-creator/
│
├── manifest.json
├── popup.html
├── popup.js
├── content.js
└── icon.png
```

---

2. High-Level Flow

1. User clicks the Chrome extension icon
2. `popup.html` opens
3. User clicks **Start Locator Mode**
4. Script is injected into the active tab
5. Hovering highlights elements
6. Clicking an element:
7. Generates locator
8. Validates uniqueness
9. Shows checkbox modal
10. User selects Playwright / Selenium / Both
11. Selected locator(s) copied to clipboard
12. Locator mode exits cleanly

---

3. manifest.json

Defines the Chrome extension configuration, permissions, and entry points.

```
{
  "manifest_version": 3,
  "name": "Locator Creator",
  "version": "1.1",
  "description": "Smart locator generator for Playwright and Selenium",
  "permissions": ["activeTab", "scripting", "clipboardWrite"],
  "action": {
    "default_popup": "popup.html"
  },
  "content_scripts": [
    {
      "matches": ["<all_urls>"],
      "js": ["content.js"]
    }
  ]
}
```

Key Permissions

- **activeTab** – Access current tab
- **scripting** – Inject JavaScript into page
- **clipboardWrite** – Copy locator to clipboard

---

4. popup.html

Defines the UI shown when the extension icon is clicked.

```
<!DOCTYPE html>
<html>
<head>
  <title>Locator Creator</title>
  <style>
    body {
      font-family: Arial;
      width: 340px;
      padding: 10px;
    }
    button {
      width: 100%;
```

```
      padding: 8px;
      cursor: pointer;
      margin-bottom: 10px;
    }
  </style>
</head>
<body>
  <h3>Locator Creator</h3>
  <button id="start">Start Locator Mode</button>
  <p>Hover and click an element</p>

  <script src="popup.js"></script>
</body>
</html>
```

---

5. content.js

Currently unused. Reserved for future enhancements like:

- Keyboard shortcuts
- Background communication
- Persistent listeners

```
// Not used currently
```

---

6. popup.js (Complete Logic)

This file:

- Injects inspection logic into the page
- Handles hover highlighting
- Generates locators
- Displays checkbox modal
- Copies selected locator(s)

```
document.getElementById("start").addEventListener("click", async () => {
  const [tab] = await chrome.tabs.query({ active: true, currentWindow: true });

  chrome.scripting.executeScript({
    target: { tabId: tab.id },
    func: startLocatorMode
  });
```

```
});

function startLocatorMode() {
  if (window.__locatorModeActive) return;
  window.__locatorModeActive = true;

  alert("Locator mode ON\nHover and click an element");

  //  🔹 Highlight box
  const highlight = document.createElement("div");
  Object.assign(highlight.style, {
    position: "absolute",
    background: "rgba(0,123,255,0.3)",
    pointerEvents: "none",
    zIndex: "999999"
  });
  document.body.appendChild(highlight);

  function move(e) {
    const r = e.target.getBoundingClientRect();
    highlight.style.top = r.top + scrollY + "px";
    highlight.style.left = r.left + scrollX + "px";
    highlight.style.width = r.width + "px";
    highlight.style.height = r.height + "px";
  }

  function click(e) {
    e.preventDefault();
    e.stopPropagation();

    // ☑️ STOP PAGE LISTENERS BEFORE SHOWING MODAL
    document.removeEventListener("mousemove", move);
    document.removeEventListener("click", click, true);

    const el = e.target;
    const result = getValidatedLocator(el);

    let playwrightCode = "";
    let seleniumCode = "";

    if (result.type === "unique") {
      playwrightCode = `page.locator("${result.locator}")`;
      seleniumCode = `driver.findElement(By.cssSelector("${result.locator}"));`;
    } else {
      playwrightCode = `page.locator("${result.locator}").nth(${result.index})`;
      seleniumCode = `driver.findElement(By.xpath("${result.xpathIndexed}"));`;
    }
```

```
  showCopyModal({ playwrightCode, seleniumCode, result });
}

document.addEventListener("mousemove", move);
document.addEventListener("click", click, true);

//  🔹 MODAL UI
function showCopyModal({ playwrightCode, seleniumCode, result }) {
  const modal = document.createElement("div");
  modal.style.cssText = `
    position: fixed;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    background: #fff;
    border-radius: 6px;
    padding: 16px;
    z-index: 1000000;
    font-family: Arial;
    box-shadow: 0 6px 20px rgba(0,0,0,0.3);
    min-width: 280px;
  `;

  modal.innerHTML = `
    <h3 style="margin-top:0">Copy Locator</h3>

    <label style="display:block;margin-bottom:6px">
      <input type="checkbox" id="pwCheck" checked />
      Playwright
    </label>

    <label style="display:block;margin-bottom:12px">
      <input type="checkbox" id="selCheck" />
      Selenium
    </label>

    <button id="copyBtn" style="
      width:100%;
      padding:8px;
      cursor:pointer;
      background:#0d6efd;
      color:#fff;
      border:none;
      border-radius:4px;
    ">
      Copy
    </button>
  `;
```

```
  // 📶 Prevent modal clicks from affecting page
  modal.addEventListener("click", e => e.stopPropagation());

  document.body.appendChild(modal);

  modal.querySelector("#copyBtn").addEventListener("click", () => {
    const copyPlaywright = modal.querySelector("#pwCheck").checked;
    const copySelenium = modal.querySelector("#selCheck").checked;

    if (!copyPlaywright && !copySelenium) {
      alert("Please select at least one option.");
      return;
    }

    let clipboardText = "";

    if (copyPlaywright) {
      clipboardText += `Playwright:\n${playwrightCode}\n\n`;
    }

    if (copySelenium) {
      clipboardText += `Selenium:\n${seleniumCode}`;
    }

    navigator.clipboard.writeText(clipboardText.trim());

    alert(
      `LOCATOR GENERATED ✅\n\n` +
      `Base Locator:\n${result.locator}\n\n` +
      `Matches Found: ${result.count}\n` +
      (result.type === "indexed"
        ? `Selected Index: ${result.index + 1}\n\n`
        : "\n") +
      `Copied:\n\n${clipboardText}`
    );

    modal.remove();
    cleanup();
  });
}

function cleanup() {
  highlight.remove();
  window.__locatorModeActive = false;
}

// ---------------- CORE FUNCTIONS ----------------
```

```
function getValidatedLocator(el) {
  const baseLocator = getBaseLocator(el);

  let elements;
  try {
    elements = Array.from(document.querySelectorAll(baseLocator));
  } catch {
    elements = [];
  }

  const count = elements.length;

  if (count === 1) {
    return { type: "unique", locator: baseLocator, count };
  }

  return {
    type: "indexed",
    locator: baseLocator,
    count,
    index: elements.indexOf(el),
    xpathIndexed: buildIndexedXPath(el)
  };
}

function getBaseLocator(el) {
  if (el.id) return `#${el.id}`;

  const testId = el.getAttribute("data-testid");
  if (testId) return `[data-testid="${testId}"]`;

  if (el.className) {
    const cls = el.className.trim().split(/\s+/).join(".");
    return `${el.tagName.toLowerCase()}.${cls}`;
  }

  return buildIndexedXPath(el);
}

function buildIndexedXPath(el) {
  let path = "";
  while (el && el.nodeType === 1) {
    let index = 1;
    let sibling = el.previousSibling;

    while (sibling) {
      if (sibling.nodeType === 1 && sibling.tagName === el.tagName) index++;
```

```
        sibling = sibling.previousSibling;
      }

      path = `/${el.tagName.toLowerCase()}[${index}]` + path;
      el = el.parentNode;
    }
    return path;
  }
}
```

> ℹ️ **Note**: The click listener is removed before showing the modal to avoid capture-phase blocking issues.

---

7. Locator Strategy

The extension generates locators using the following priority:

1. **ID selector**

```
#submitBtn
```

1. **data-testid**

```
[data-testid="login-button"]
```

1. **Tag + class**

```
button.primary.large
```

1. **Fallback XPath (indexed)**

```
/html/body/div[1]/form[1]/button[2]
```

---

8. Playwright vs Selenium Output

Playwright

```
page.locator("#login")
page.locator(".btn.primary").nth(1)
```

Selenium

```
driver.findElement(By.cssSelector("#login"));
driver.findElement(By.xpath("/html/body/..."));
```

---

9. Modal UI Behavior

- Two checkboxes:
- Playwright (default checked)
- Selenium
- User may select one or both
- Copy button copies selected locator(s)
- Validation ensures at least one option is selected

---

10. Cleanup & Safety

After copying:

- Mouse listeners removed
- Highlight overlay removed
- Modal removed
- Locator mode flag reset

This ensures **no side effects on the page**.

---

11. Known Non-Issues

Console messages like:

- Chrome third-party cookie warnings
- Microsoft Clarity logs

Are **from the website**, not the extension.

---

12. Future Enhancements

Recommended next steps:

- ESC key to close modal
- Dark mode modal

- Remember last selection
- Locator history panel
- Playwright `getByRole`, `getByText`
- Relative XPath generation
- DevTools panel integration

---

13. Summary

This extension provides a **production-ready locator generation workflow** for automation engineers:

✔️Visual inspection ✔️Smart locator logic ✔️Multi-framework support ✔️Clean UX ✔️Safe DOM handling

---

Happy Automating 🚫