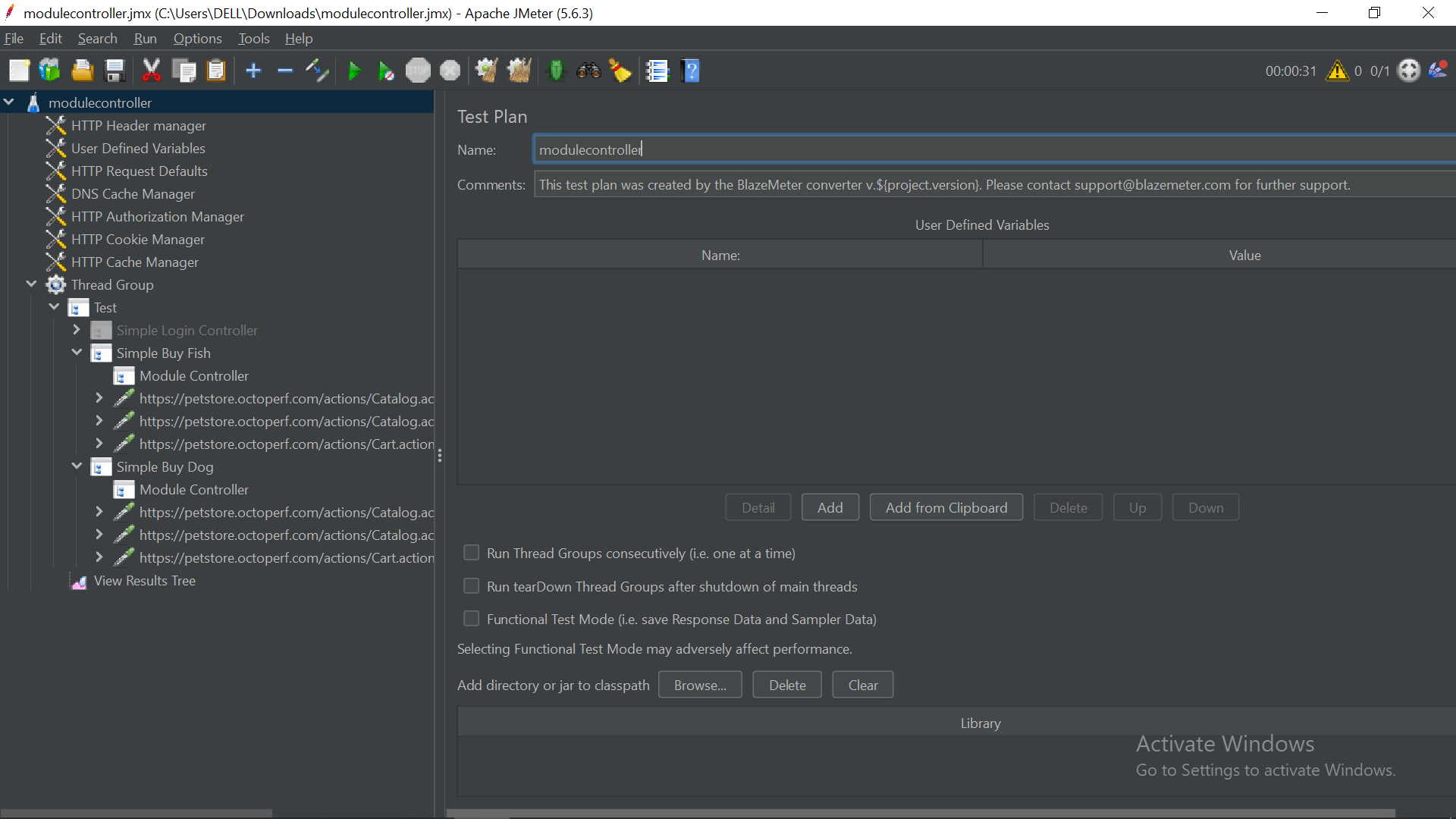
🧩 Module Controller





The **Module Controller** is a powerful **Logic Controller** in JMeter that enables **modularization** and **reusability** of test elements. It works similarly to a **function call** in programming, allowing you to define a sequence of actions once and reuse them across your test plan.

**🧠 Why Use a Module Controller?**

In large-scale or complex test plans, you often need to repeat the same group of requests (e.g., login, search, logout). Instead of copying and pasting those steps multiple times, the **Module Controller lets you define them once and reference them wherever needed**.

**⚙️ How It Works**

**Step 1: Define the Module**

* Create a **Test Fragment** (or a Simple Controller) in your Test Plan.
* Place your reusable logic inside it (e.g., Login Sequence, Search Flow).

**Step 2: Add a Module Controller**

* In your **Thread Group**, insert a **Module Controller**.
* In the "Module to Run" dropdown, select the **controller** that contains your reusable logic.

✅ When the test runs, the Module Controller **executes all child elements** under the selected controller.

**🧪 Example Structure**

java

CopyEdit

Test Plan

├── Test Fragment

│ └── Simple Controller (Login\_Module)

│ ├── HTTP Request (GET Login Page)

│ └── HTTP Request (POST Credentials)

│

└── Thread Group (User Scenario)

├── Module Controller (Runs Login\_Module)

│

├── Simple Controller (Browse\_Products)

│ ├── HTTP Request (View All Products)

│ └── HTTP Request (View Product Details)

│

├── Module Controller (Runs Login\_Module)

│

└── Simple Controller (Checkout)

├── HTTP Request (Add to Cart)

└── HTTP Request (Complete Purchase)

🟢 In this structure, **Login\_Module** is defined once but reused multiple times via the **Module Controller**.

**🧰 Key Benefits**

| **Benefit** | **Description** |
| --- | --- |
| **Reusability** | Define once, reuse many times—avoids duplication. |
| **Maintainability** | Update logic in one place—changes reflect wherever it’s reused. |
| **Readability** | Keeps your test plan clean and logical—makes the user flow more understandable. |
| **Modularity** | Encourages structured and component-based design for test cases. |

**⚖️ Module Controller vs. Include Controller**

| **Feature** | **Module Controller** | **Include Controller** |
| --- | --- | --- |
| Scope | Same JMX file | Different (external) JMX files |
| Usage | Reuse parts defined **within** the test plan | Reuse **external** test fragments |
| Configuration | Select module from dropdown menu | Provide file path to include |
| Ideal For | Modularization inside one plan | Sharing logic across multiple test plans |

**✅ Best Practices**

* ✅ Use **Test Fragments** to host reusable modules.
* ✅ Clearly name each module (e.g., Login\_Module, Search\_Module).
* ✅ Avoid circular references to prevent infinite loops.
* ✅ Keep your modules **self-contained** with their own assertions, timers, and processors if needed.