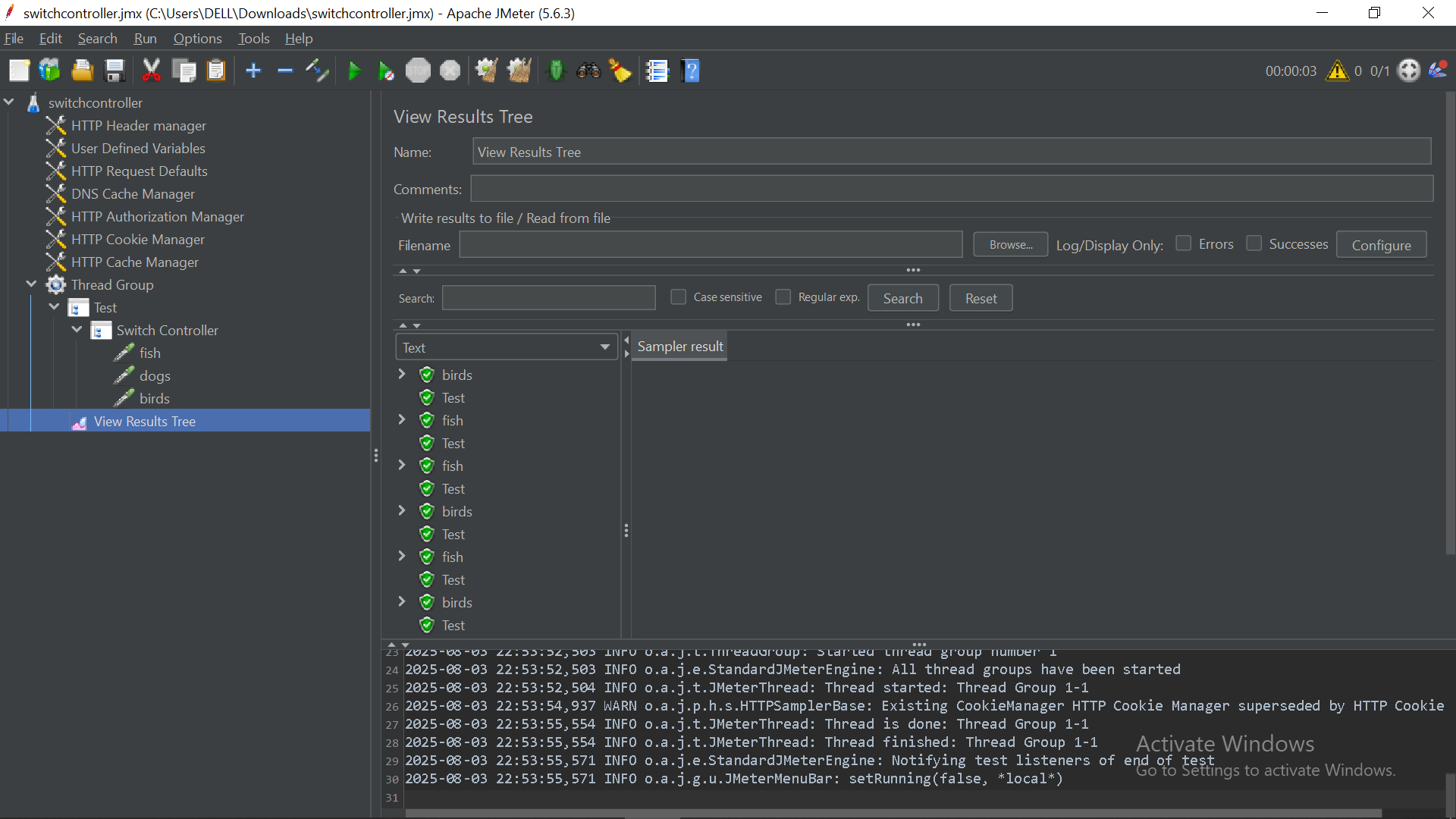
🔀 Switch Controller



The **Switch Controller** is a powerful **Logic Controller** in JMeter that allows you to **conditionally execute one child element** based on a specified **index or string value**—much like a switch or case statement in programming.

This is ideal for simulating **diverse user behavior** or controlling the test flow dynamically based on **variables**, **functions**, or **runtime conditions**.

**🧠 When to Use**

Use the Switch Controller when you want to:

* ✅ Simulate random user actions (browse, add to cart, checkout).
* ✅ Route logic based on response values or extracted variables.
* ✅ Trigger different flows based on command-line arguments or test parameters.
* ✅ Reduce duplication by conditionally executing one of several paths.

**⚙️ How It Works**

The controller evaluates the value in the **Switch Value** field and executes:

* The **child element with a matching index**, starting from 0, OR
* The **child element with a matching name**.

If no match is found, **nothing is executed**—unless a **Default Switch Value** is configured.

**🧰 Key Configuration Options**

| **Setting** | **Description** |
| --- | --- |
| **Switch Value** | Expression to evaluate—can be an index (0, 1), a string (browse), a variable (${action}), or a function (${\_\_Random(0,2,)}) |
| **Default Value** | Optional fallback—executes if no match is found for the primary Switch Value |

**📦 Example 1: Using Index (Number-Based Switching)**

**🧪 Scenario: Randomly simulate one of three user actions.**

**Setup:**

1. **Switch Value:** ${\_\_Random(0,2,)}
2. **Child Controllers:**
   * 0\_Browse\_Products
   * 1\_Add\_to\_Cart
   * 2\_Checkout

**Behavior:**

On each iteration, a random number (0–2) is generated. The corresponding child controller is executed.

java

CopyEdit

Test Plan

└── Thread Group

├── Switch Controller (Switch Value: ${\_\_Random(0,2,)})

│ ├── Simple Controller (Name: 0\_Browse\_Products)

│ ├── Simple Controller (Name: 1\_Add\_to\_Cart)

│ └── Simple Controller (Name: 2\_Checkout)

**📦 Example 2: Using Names (String-Based Switching)**

**🧪 Scenario: More readable version using named actions.**

**Setup:**

1. Add a **JSR223 PreProcessor** to set a variable:

groovy

CopyEdit

vars.put("action\_name", ["browse", "add", "checkout"][new Random().nextInt(3)])

1. **Switch Controller:**
   * Switch Value: ${action\_name}
2. **Child Controllers (named to match):**
   * browse
   * add
   * checkout

**Behavior:**

One of the named controllers is selected and executed based on the variable action\_name.

java

CopyEdit

Test Plan

└── Thread Group

├── JSR223 PreProcessor (sets ${action\_name})

├── Switch Controller (Switch Value: ${action\_name})

│ ├── Simple Controller (Name: browse)

│ ├── Simple Controller (Name: add)

│ └── Simple Controller (Name: checkout)

**✅ Best Practices**

* Prefer **named switching** for better readability and flexibility.
* Use the **Default Switch Value** as a safety net to avoid silent failures.
* Combine with **User Defined Variables**, **JSR223**, or **PostProcessors** to create context-aware dynamic flows.
* Avoid overly complex logic inside child controllers to maintain clarity.

**⚠️ Common Pitfalls**

| **Pitfall** | **Solution** |
| --- | --- |
| Using string names without matching controller names | Ensure child controller names exactly match the switch value |
| No match found, and no fallback | Set a **Default Switch Value** if execution must continue |
| Index-based switch breaks when order changes | Prefer name-based switching for maintainability |

**📌 Summary**

| **Feature** | **Description** |
| --- | --- |
| Type | Logic Controller |
| Executes | Only one matching child element (by index or name) |
| Supports | Variables, Functions, Static values |
| Use Case Examples | Random action simulation, routing based on condition |
| Alternative | **If Controller** (for multiple condition branches) |