

# Code Refactoring Summary: spip.cpp

Antigravity Agent

January 29, 2026

## 1 Overview

Following alpha-equivalence analysis and pattern detection, we identified and refactored several common code patterns in `spip.cpp` to improve maintainability and reduce code duplication.

## 2 Refactorings Applied

### 2.1 1. Site-Packages Directory Search (Already Completed)

**Pattern:** Recursive directory iteration to find `site-packages`

**Occurrences:** 8 locations

**Solution:** Extracted into helper function `get_site_packages(const Config& cfg)`

```
fs::path get_site_packages(const Config& cfg) {
    if (!fs::exists(cfg.project_env_path)) return fs::path();
    try {
        for (const auto& entry : fs::recursive_directory_iterator(
            cfg.project_env_path)) {
            if (entry.is_directory() &&
                entry.path().filename() == "site-packages") {
                return entry.path();
            }
        }
    } catch (...) {}
    return fs::path();
}
```

### 2.2 2. Command Argument Validation (New)

**Pattern:** Check argument count and print usage message

**Occurrences:** 7 locations

**Solution:** Extracted into helper function `require_args`

```
bool require_args(const std::vector<std::string>& args,
                  size_t min_count,
                  const std::string& usage_msg) {
    if (args.size() < min_count) {
        std::cout << usage_msg << std::endl;
        return false;
    }
    return true;
}
```

**Before:**

```
if (command == "bundle") {
    if (args.size() < 2) {
        std::cout << "Usage: spip bundle <folder>" << std::endl;
        return;
    }
}
```

```
    bundle_package(cfg, args[1]);
}
```

**After:**

```
if (command == "bundle") {
    if (!require_args(args, 2, "Usage: spip bundle <folder>"))
        return;
    bundle_package(cfg, args[1]);
}
```

## 2.3 3. Setup + Execute Pattern (New)

**Pattern:** setup\_project\_env(cfg); function\_name(cfg);

**Occurrences:** 4 locations (prune, audit, review, verify)

**Solution:** Extracted into higher-order helper function `exec_with_setup`

```
void exec_with_setup(Config& cfg,
                    std::function<void(Config&)> func) {
    setup_project_env(cfg);
    func(cfg);
}
```

**Before:**

```
else if (command == "prune") {
    setup_project_env(cfg);
    prune_orphans(cfg);
}
else if (command == "audit") {
    setup_project_env(cfg);
    audit_environment(cfg);
}
```

**After:**

```
else if (command == "prune") {
    exec_with_setup(cfg, prune_orphans);
}
else if (command == "audit") {
    exec_with_setup(cfg, audit_environment);
}
```

## 3 Impact

Refactoring	Lines Saved	Occurrences
get_site_packages	~32	8
require_args	~21	7
exec_with_setup	~8	4
<b>Total</b>	<b>~61</b>	<b>19</b>

Table 1: Code reduction metrics

## 4 Benefits

- **Reduced Duplication:** Eliminated 19 instances of repeated code patterns

- **Improved Maintainability:** Bug fixes and enhancements now require changes in only one location
- **Clearer Intent:** Higher-level abstractions make the code more readable
- **Type Safety:** The `std::function` template ensures compile-time type checking

## 5 Verification

All refactorings were verified by:

- Successful compilation with `clang++ -std=c++20`
- Testing argument validation (`./spip bundle` returns correct usage message)
- Ensuring command dispatch still functions correctly