#Hack

PROGRAMMING LANGUAGE

FROM BEGINNER TO EXPERT:
A COMPREHENSIVE GUIDE TO MODERN HACK DEVELOPMENT

START STRONG
BUILD A SOLID FOUNDATION FROM THE GROUND UP

WRITE MODERN CODE
EXPLORE HACK'S POWERFUL FEATURES AND TYPE SYSTEM

ADVANCED CONCEPTS
ASYNC PROGRAMMING, GENERICS, SHAPES AND MORE

PRODUCTION READY
ARCHITECT, TEST AND DEPLOY APPLICATIONS AT SCALE

MASON BROOKS

Hack Programming Language

From Beginner to Expert: A Comprehensive Guide to Modern Hack Development

Steve Publications

This book is available at https://leanpub.com/hackprogramminglanguage

This version was published on 2026-08-03

Contents

From Beginner to Expert: A Comprehensive Guide to Modern Hack Development

Hack is a modern programming language developed by Meta that combines the rapid development cycle of dynamic scripting with the reliability and performance of static typing. This book takes you on a complete journey from your first lines of code through advanced concepts like asynchronous programming, generics, shapes and production deployment. Whether you are migrating from PHP or approaching Hack for the first time, this guide provides thorough explanations, production-ready code examples and practical architectural patterns used by engineers building systems at scale. By the end, you will understand not only how to write Hack code but why it is designed the way it is, enabling you to build reliable, high-performance applications with confidence.

Introduction

The Problem Hack Solves

Imagine you are building a web application that starts small. A few endpoints, some database queries, templates rendering HTML to browsers. You choose PHP because it is familiar, it runs everywhere, and you can prototype quickly. This is exactly how Facebook started. Mark Zuckerberg wrote the first version of Facebook in PHP, and for years it served the platform well.

Then the user base grew. From thousands to millions to billions. The codebase followed suit, swelling to tens of millions of lines. Bugs that were once harmless edge cases became production incidents affecting hundreds of millions of users. A missing type annotation led to a null reference somewhere deep in the call stack. An unescaped string opened a cross-site scripting vulnerability. A subtle race condition corrupted data under heavy load.

The team needed change, but rewriting the entire platform in a new language was not realistic. They needed something that would work with the existing codebase while providing the safety and performance characteristics of a modern systems language. That need gave birth to Hack.

What Hack Is (and Is Not)

Hack is not just PHP with type hints slapped on top. It is not a library or a framework. Hack is a distinct programming language that shares PHP ancestry but has evolved into something fundamentally different.

At its core, Hack reconciles two competing goals in software development:

- **Developer velocity**: The ability to write code quickly, iterate rapidly and ship features without bureaucratic overhead.
- **Code reliability**: The assurance that your code will behave correctly under load, handle edge cases gracefully and resist bugs introduced by refactoring.

Traditional dynamically typed languages excel at velocity but struggle with reliability at scale. Traditional statically typed languages guarantee reliability but can slow development with verbose boilerplate and complex build processes. Hack aims for both.

Hack achieves this through several key design decisions:

Gradual typing: You can start writing code without any type annotations and gradually add them as your codebase matures. Typed and untyped code interoperate seamlessly, allowing incremental migration of large existing projects.

Instantaneous type checking: Unlike many statically typed languages that require lengthy compilation steps, Hack's type checker runs incrementally in under 200 milliseconds, providing feedback as you type without disrupting your workflow.

Runtime enforcement: When you add a type annotation to a function parameter or return value, Hack enforces it at runtime. This catches bugs even before you have fully typed your codebase, because dynamic code calling typed code must respect the declared types.

Modern language features: Hack includes generics, traits, enums, async/await, lambda expressions and shape types , features that many modern languages take for granted but were absent from PHP when Hack was designed.

The Journey Ahead

This book is structured as a progressive learning journey. We begin with history and context so you understand why Hack exists and what problems it is designed to solve. Then we move through installation and your first programs, building up the fundamental syntax and concepts you need to write correct code.

As you progress, each chapter builds on the previous ones:

- **Foundations** (Chapters 1–4): History, setup, basic syntax, variables, types, operators and control flow.
- **Core language features** (Chapters 5–7): Functions, collections and Hack's powerful type system with generics and shapes.

- **Object-oriented programming** (Chapters 8–9): Classes, inheritance, interfaces, traits and enums.
- **Organization and structure** (Chapter 10): Namespaces, modules and how to organize large codebases.
- **Robustness** (Chapters 11–13): Error handling, async programming, attributes and reflection.
- **Professional development** (Chapters 14–17): Testing, debugging, performance optimization, PHP migration, architecture patterns, security and production deployment.

Every concept is explained clearly before diving into technical detail. Every code example is complete, runnable and designed to be production-quality rather than contrived. When multiple approaches exist, I explain the trade-offs so you can make informed decisions in your own projects.

Who This Book Is For

This book assumes you are a software developer with some programming experience, but it does not assume any prior knowledge of Hack or PHP. If you have written code in any mainstream language (Python, JavaScript, Java, C#, Go, or yes even PHP), you will be able to follow along.

If you are a PHP developer considering migration to Hack, this book will show you both the similarities and the important differences, helping you avoid common pitfalls when converting existing code.

If you are evaluating Hack for use in your organization, the later chapters on architecture, deployment and production practices will give you the practical information needed to make an informed decision.

How to Use This Book

Read sequentially if you are new to Hack. The chapters are ordered to build understanding progressively, and skipping ahead may leave gaps in your knowledge.

Use it as a reference if you already know Hack but need a reliable source for specific topics like generics variance, async error handling, or shape type constraints.

Experiment with every code example. Hack is best learned by writing code, running it, breaking it and fixing it. The examples are designed to be modified and extended as you learn.

Now let us begin with the story of how Hack came to exist, because understanding its history reveals much about its design philosophy and strengths.

Chapter 1: The Story of Hack: History, Philosophy and Design Goals

The PHP Problem: Scaling a Billion-Person Platform

In 2004, when Facebook launched, it was built entirely in PHP. This was a natural choice. PHP was designed for web development from day one, had excellent hosting support and allowed developers to ship features quickly. For a small team building a social network for college students, PHP was perfect.

By 2010, Facebook had grown into one of the largest websites in the world, with hundreds of millions of users and a codebase that had expanded to tens of millions of lines of code. The problems that come with this scale were no longer theoretical.

Dynamic typing, which had been a convenience during the early days, became a liability. Without compile-time type checking, bugs slipped through until they hit production. A function expecting an integer might silently receive null, causing failures deep in the call stack that were difficult to trace. Refactoring large sections of code was risky because there was no automated way to verify that all callers still matched updated signatures.

Performance was another concern. The standard PHP interpreter (the Zend Engine) parses and compiles scripts on every request. For a site serving billions of requests per day, this overhead added up significantly. Facebook engineers needed something faster.

HipHop for PHP: The First Attempt

Facebook's first major response to the performance problem was HipHop for PHP (HPHPc), open-sourced in 2010. HPHPc was a transpiler that converted PHP source code into C++, which was then compiled to native machine code. This approach doubled Facebook's server capacity and reduced CPU usage dramatically.

However, HPHPc had serious limitations:

- **Long compile times**: Transpiling millions of lines of PHP to C++ and then compiling the result took hours. Every deployment required this lengthy process.
- **Incomplete language support**: Not all PHP features could be cleanly translated to C++. Some dynamic behaviors that were essential to Facebook's codebase could not be supported.
- **Difficult debugging**: When something went wrong, the stack traces pointed to generated C++ code rather than the original PHP, making diagnosis painful.
- **No incremental compilation**: Any change required recompiling the entire codebase, which was impractical for a team shipping changes multiple times per day.

HPHPc proved that compiling PHP to native code could deliver massive performance gains, but it also showed that transpilation alone was not a sustainable long-term solution.

Birth of HHVM: A JIT Compiler for the Real World

Around 2010, Facebook began work on HipHop Virtual Machine (HHVM), released in December 2011. HHVM took a fundamentally different approach from HPHPc.

Instead of transpiling PHP to C++ ahead of time, HHVM used just-in-time (JIT) compilation. When Hack or PHP code runs on HHVM, it is first compiled into an intermediate bytecode format called HipHop Bytecode (HHBC). This bytecode is then interpreted initially, but as the JIT compiler identifies frequently executed code paths ("hot" code), it compiles those sections directly to optimized x86-64 machine code at runtime.

This approach solved HPHPc's biggest problems:

- **Fast startup**: No lengthy pre-compilation step was needed. HHVM could start serving requests immediately.
- **Incremental optimization**: Only the hot paths were compiled, so the JIT focused its effort where it mattered most.

- **Better debugging**: Stack traces could map back to the original source code more reliably.

By Q1 2013, Facebook had migrated its production website from HPHPc to HHVM, marking a major milestone in the platform's evolution.

The story of how HHVM came to exist is itself interesting. According to Keith Adams, one of the engineers involved, the team initially attempted to run PHP code through Google's V8 JavaScript engine. They hit fundamental incompatibilities, particularly around garbage collection strategies, and decided instead to build their own JIT compiler from scratch. That decision led directly to HHVM and eventually to Hack itself.

Hack Emerges: From XHP to a New Language

HHVM was originally designed primarily as a PHP runtime, but Facebook engineers recognized that to fully exploit the JIT compiler's capabilities and to address the type safety problems of their massive codebase, they needed more than just a faster interpreter. They needed language-level changes.

The journey toward Hack began with XHP, an extension to PHP syntax that allowed HTML and XML fragments to be written as first-class expressions. XHP provided type-checked HTML generation that automatically prevented cross-site scripting (XSS) vulnerabilities by properly escaping all output. It was a success internally and demonstrated the value of adding structured, type-safe features to the PHP ecosystem.

Building on this experience, Julien Verlaguet, Alok Menghrajani, Drew Paroski and their colleagues at Facebook designed Hack as a new programming language that would:

- Interoperate seamlessly with existing PHP code
- Add static typing without sacrificing development velocity
- Include modern language features missing from PHP
- Optimize for HHVM's JIT compilation model

Hack was announced and open-sourced on March 20, 2014. The announcement blog post stated clearly: "Hack reconciles the fast development cycle of PHP with the discipline provided by static typing, while adding many features commonly found in other modern programming languages."

Within a year of its release, Facebook had migrated nearly its entire PHP codebase to Hack. This was not accomplished through a big-bang rewrite but through gradual, incremental migration, exactly the kind of approach that Hack's design philosophy encourages.

Design Philosophy: Pragmatism Over Purity

Hack's design decisions reflect a pragmatic engineering mindset shaped by the realities of maintaining one of the world's largest codebases. Several principles guide its design:

Gradual typing as a first-class feature

Hack is a gradually typed language, meaning that static and dynamic typing coexist in the same program. Functions can have typed parameters and return values while other functions remain completely untyped. Typed code calling untyped code trusts the dynamic types at runtime, while untyped code calling typed code must respect the declared type contracts.

This design enables incremental migration. A team can start adding type annotations to their most critical or error-prone code first, then gradually expand coverage over time. There is no requirement to type everything before getting any benefit from the type system.

Runtime enforcement of type annotations

When you add a type annotation to a Hack function, that type is enforced at runtime by HHVM, not just checked statically by the type checker. This means that even before your entire codebase is fully typed, the runtime will catch type violations where typed and untyped code interact.

This design choice serves two purposes:

1. It provides safety during migration, catching bugs that static analysis might miss because it cannot see all the dynamic code paths.
2. It enables the JIT compiler to optimize more aggressively, because it can trust that values passed through typed boundaries actually have the declared types.

Instantaneous feedback

Hack's type checker is designed to run incrementally as you edit files, providing feedback in under 200 milliseconds. This is critical for developer productivity. If type checking takes too long, developers will disable it or work around it, defeating its purpose. Hack's designers understood that the best type system is one that developers actually use.

Influences from many languages

Hack draws inspiration from a wide range of languages, including PHP (its ancestor), OCaml (for type system concepts), Java and C# (for object-oriented features and generics), Scala (for functional programming elements) and Haskell (for type-level abstractions). This eclectic borrowing reflects Hack's pragmatic philosophy: if a feature works well in another language and would benefit Hack developers, it is worth considering.

No HTML mixing

Unlike PHP, Hack does not allow embedding code within HTML templates using the traditional `<?php ?>` tags. All Hack code must be written in pure `.hack` files, and HTML generation should use XHP or other templating approaches. This separation of concerns makes the codebase cleaner and more maintainable, even if it represents a break from PHP tradition.

Hack vs. PHP: Understanding the Relationship

Hack shares its ancestry with PHP, but they have diverged significantly over time. Understanding their relationship is important for anyone coming from either language.

What Hack inherited from PHP

- Basic syntax: variable names with `$`, operators like `+`, `-`, `==`, `===`, control flow statements like `if`, `while`, `for`, `foreach`
- Familiar concepts: arrays (though reimagined), functions, classes, namespaces
- The overall feel of a scripting language optimized for web development

What Hack changed or removed

- No HTML embedding: Hack files contain only code, not mixed markup and logic.
- No global variables or superglobals: Variables are scoped to functions and classes.
- Stricter type conversion rules: An integer is not automatically coerced to a float or string.
- No variable variables: The PHP pattern of `$$varname` is disallowed because it breaks static analysis.
- Functions must be declared inside namespaces or use explicit entry points.

What Hack added

- Static type annotations for parameters, return values and class properties
- Generics for type-safe collections and functions
- Shape types for typed associative arrays
- Traits for horizontal code reuse
- Native enum support
- Async/await for non-blocking I/O
- Lambda expressions with concise syntax (==> fat arrow)
- Attributes (annotations) for metadata

The fork: HHVM ending PHP support

In September 2018, the HHVM team announced that starting with version 4.0, HHVM would no longer aim to support the PHP language. This decision reflected the growing divergence between Hack and PHP and the realization that maintaining compatibility with both languages was slowing progress on features that specifically benefited Hack users.

HHVM 3.30 was the final release series supporting PHP, with end of support in November 2019. Users who wanted to continue using HHVM had to migrate their code to Hack or switch to a traditional PHP runtime like the Zend Engine.

This decision was controversial but consistent with Hack's design philosophy: optimization for real-world use at scale takes priority over maintaining compatibility for its own sake.

Hack vs. TypeScript: A Common Comparison

TypeScript is often compared to Hack because both languages add static typing to an existing dynamically typed language (JavaScript and PHP respectively) and both support gradual typing. However, they have important differences:

Aspect	Hack	TypeScript
Runtime	HHVM (JIT compiled)	JavaScript engine (interpreted/JIT)
Type checking	Built-in type checker (`hh_client`)	Compiler (`tsc`)
Runtime enforcement	Types enforced at runtime by HHVM	Types erased at compile time
Async model	Native async/await with cooperative multitasking	Promises-based (language-level syntax, runtime-dependent)
Collections	Typed vec, dict, keyset with value semantics	Untyped arrays and objects
HTML integration	XHP for type-safe templating	JSX for React components

TypeScript is a transpiler that compiles to JavaScript and runs on any JavaScript engine. Hack is a distinct language that runs only on HHVM. TypeScript's types are erased at compile time, while Hack's types are enforced both statically and at runtime.

The Modern Hack Ecosystem

As of 2025, Hack continues under active maintenance at Meta, with daily internal commits and periodic synchronization to the public HHVM repository on GitHub. Meta uses hundreds of millions of lines of Hack code in production, powering core services including Facebook, Instagram and WhatsApp's backend infrastructure.

The broader ecosystem has evolved:

- **Tooling**: The Hack toolchain includes `hh_client` for type checking, `hackfmt` for formatting, HHAST for linting and migrations and editor integrations for VS Code, Vim and Emacs.
- **Testing**: HackTest (released in 2018) provides a native unit testing framework that does not depend on PHP's PHPUnit.
- **Standard library**: The Hack Standard Library (HSL) provides well-typed, idiomatic implementations of common operations for collections, strings, math and more.
- **Community adoption**: While external adoption has been limited compared to PHP or TypeScript, organizations like Slack have successfully migrated large codebases to Hack and reported significant benefits in developer productivity and code reliability.

In May 2025, Meta announced full-stack HHVM optimizations tailored for generative AI workloads, demonstrating that Hack and HHVM continue to evolve alongside emerging technology trends.

Why Hack Still Matters

You might wonder: if PHP 7 and 8 have incorporated many features inspired by Hack (type declarations, JIT compilation, improved error handling), is Hack still relevant?

The answer depends on your context. For small projects or teams already comfortable with modern PHP, the benefits of migrating to Hack may not justify the effort. PHP has become a much better language than it was in 2014.

But for large-scale systems where:

- Type safety at runtime matters as much as static type checking
- The incremental type checking feedback loop is critical for developer velocity
- Async I/O without multithreading complexity is valuable
- XHP provides safer HTML generation at scale

Hack remains a uniquely capable tool. It was designed for the specific challenges of building reliable software at epic scale, and it continues to serve that purpose effectively.

Even if you never write Hack code professionally, studying it offers valuable insights into language design, gradual typing and the trade-offs between developer productivity and system reliability. Many concepts from Hack have influenced other languages, and understanding Hack helps you understand the broader landscape of modern programming language evolution.

Chapter Summary

- Hack was created by Meta to address type safety and performance challenges in their massive PHP codebase.
- HHVM, a JIT-compiled runtime, replaced the earlier HPHPc transpiler and provided the foundation for Hack.
- Hack's design philosophy prioritizes pragmatism: gradual typing, runtime enforcement, instantaneous feedback and influences from many languages.
- Hack has diverged significantly from PHP and HHVM no longer supports PHP as of version 4.0.
- Hack continues to be actively maintained and used at Meta, with recent optimizations for AI workloads.
- Understanding Hack's history reveals why it is designed the way it is and when it is the right tool for the job.

In the next chapter, we will set up your development environment and write your first Hack programs.

Chapter 2: Getting Started: Installation, Environment and Your First Program

System Requirements and Platform Support

Before installing Hack and HHVM, let us review the platform requirements.

HHVM officially supports Linux distributions based on Ubuntu and Debian. The primary supported architectures are x86-64 (64-bit Intel and AMD processors). While community efforts have produced builds for other platforms, official support is focused on Linux because that is where Meta deploys HHVM in production.

Key requirements:

- **Operating system**: Ubuntu 20.04 or later, Debian 10 or later (officially supported)
- **Architecture**: x86-64
- **Memory**: At least 2 GB RAM recommended for development; more for large codebases
- **Disk space**: Approximately 500 MB for HHVM plus dependencies
- **Development tools**: Build essentials if compiling from source

HHVM does not officially support Windows or macOS as primary development platforms. However, you can run HHVM in Docker containers on any operating system that supports Docker, which is the recommended approach for non-Linux environments.

Installing HHVM: Packages and Methods

There are several ways to install HHVM depending on your needs and environment. Let us cover the most common approaches.

Method 1: Prebuilt Packages (Recommended)

The simplest way to get started is to use prebuilt packages from the official HHVM repository. This approach gives you a stable, tested installation without needing to compile anything.

For Ubuntu, add the HHVM repository and install:

```
sudo apt-get update
sudo apt-get install -y apt-transport-https software-properties-common
sudo apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80
↪  0xB4112585D386EB94
sudo add-apt-repository https://dl.hhvm.com/ubuntu
sudo apt-get update
sudo apt-get install hhvm
```

For Debian:

```
sudo apt-get update
sudo apt-get install -y apt-transport-https software-properties-common
sudo apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80
↪  0xB4112585D386EB94
sudo add-apt-repository https://dl.hhvm.com/debian
sudo apt-get update
sudo apt-get install hhvm
```

After installation, verify that HHVM is working:

```
hhvm --version
```

You should see output showing the HHVM version number. If you see a version string, your installation was successful.

Method 2: Docker Containers

If you are on macOS, Windows, or prefer containerized development, Docker provides an excellent way to run HHVM without installing it directly on your host system.

Official HHVM Docker images are available on Docker Hub:

```
docker pull hhvm/hhvm
```

You can then run a Hack file inside the container:

```
docker run --rm -v $(pwd):/app -w /app hhvm/hhvm hhvm hello.hack
```

For more sophisticated development, create a Docker Compose configuration that mounts your project directory and provides a consistent development environment:

```
version: '3.8'
services:
  hack-dev:
    build: .
    volumes:
      - .:/app
    working_dir: /app
    command: ["tail", "-f", "/dev/null"]
```

With a corresponding Dockerfile:

```
FROM hhvm/hhvm
RUN apt-get update && apt-get install -y git curl
WORKDIR /app
```

This setup lets you run HHVM commands inside the container while editing files on your host system.

Method 3: Building from Source

Building from source gives you access to the latest features and allows customization, but it requires more time and disk space. This approach is recommended primarily for contributors to the HHVM project or developers who need specific configuration options not available in prebuilt packages.

Basic steps:

```
sudo apt-get update
sudo apt-get install software-properties-common apt-transport-https
sudo apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80
↪  0xB4112585D386EB94
sudo add-apt-repository https://dl.hhvm.com/ubuntu
sudo apt-get update
sudo apt-get build-dep hhvm-nightly

git clone https://github.com/facebook/hhvm.git
cd hhvm
./configure
make -j$(nproc)
sudo make install
```

Building from source can take a significant amount of time (often an hour or more depending on your hardware). For most developers, prebuilt packages are the better choice.

Setting Up the Hack Typechecker

HHVM includes the Hack typechecker as `hh_client`. This tool performs static analysis on your codebase and reports type errors before you run your program. Setting it up correctly is essential for getting the full benefit of Hack's type system.

Creating a Project Configuration File

Every Hack project needs an `.hhconfig` file in its root directory. This file configures the typechecker behavior. Create a basic configuration:

```
; .hhconfig
typechecker =
  strictness_consistency=true
  suppress_errors_from=/vendor/
```

Key configuration options:

- **strictness_consistency**: Ensures that files within the same directory use consistent strictness modes.
- **suppress_errors_from**: Tells the typechecker to ignore errors in specified directories (typically vendor dependencies).

For more advanced projects, you may want to add:

```
typechecker =
  strictness_consistency=true
  suppress_errors_from=/vendor/
  allow_inconsistent_strictness=false
  enable_linter=true
```

Running the Typechecker

Start the typechecker daemon in your project directory:

```
hh_client --watch
```

The `--watch` flag keeps the daemon running and automatically re-checks files as you edit them. This provides near-instant feedback during development.

To run a one-time check:

```
hh_client
```

This scans all Hack files in your project and reports any type errors found.

Project Structure and Configuration Files

A well-organized project structure makes your codebase easier to navigate and maintain. Here is a recommended structure for a typical Hack project:

```
my-project/
├── .hhconfig              ; Typechecker configuration
├── composer.json          ; Dependency management
├── hh_autoload.json       ; HHVM autoloading configuration
├── src/                   ; Application source code
│   ├── Main.hack
│   └── Services/
│       └── UserService.hack
├── tests/                 ; Test files
│   └── UserServiceTest.hack
└── vendor/                ; Dependencies (gitignored)
```

Composer Configuration

Hack projects use Composer for dependency management, just like PHP projects. Create a `composer.json`:

```
{
    "name": "my-org/my-project",
    "require": {
        "hhvm/hhvm-autoload": "^1.0",
        "hhvm/hsl": "*"
    },
    "require-dev": {
        "hhvm/hacktest": "*",
        "facebook/fbexpect": "*"
    },
    "autoload": {
        "psr-4": {
            "MyProject\\": "src/"
        }
    }
}
```

Install dependencies:

```
composer install
```

Autoloading Configuration

HHVM uses an autoloader to find and load classes as they are referenced. Create `hh_autoload.json`:

```
{
    "roots": [
        {
            "path": "src",
            "namespace": "MyProject"
        }
    ],
    "devRoots": [
        {
            "path": "tests",
            "namespace": "MyProject\\Tests"
        }
    ]
}
```

Generate the autoload map:

```
vendor/bin/hh-autoload
```

This creates a mapping file that HHVM uses to locate classes by their fully qualified names.

Your First Hack Program: Hello World with Explanations

Now let us write and run your first Hack program. Create a file called `hello.hack`:

```
<?hh // strict

namespace MyProject;

<<__EntryPoint>>
function main(): void {
    echo "Hello, World!\n";
}
```

Let us break down every line of this program:

`<?hh // strict`: This is the Hack file header. The `<?hh` opening tag identifies this as a Hack file (as opposed to PHP's `<?php`). The `// strict` comment specifies that this file uses strict mode, which enforces the strongest type checking rules. There are three possible modes:

- **strict** (recommended): Full static type checking with all modern Hack features. This is the default for new code.
- **partial**: Relaxed type checking designed to ease migration from PHP. Untyped values are treated as `any` type. Discouraged for new code.
- No mode specified: Behavior depends on HHVM version and configuration. Avoid this ambiguity.

`namespace MyProject;`: This declares that all code in this file belongs to the `MyProject` namespace. Namespaces organize code into logical groups and prevent naming collisions. Every Hack file should declare a namespace, and you cannot have top-level code outside of namespaces (unlike PHP).

`<<__EntryPoint>>`: This is an attribute (also called an annotation) that marks this function as the entry point of the program. Hack requires explicit entry points because it does not allow top-level executable statements. When HHVM runs a file, it looks for a function marked with `<<__EntryPoint>>` and calls it automatically.

`function main(): void { ... }`: This declares a function named `main`. The `: void` after the parameter list is a return type annotation specifying that this function does not return a value. In Hack, you can annotate function parameters and return values with types, and these annotations are enforced both statically by the typechecker and at runtime by HHVM.

`echo "Hello, World!\n";`: This outputs text to standard output. The `\n` is a newline character. Note that Hack uses double quotes for strings with escape sequences, just like PHP.

Running Your First Program

Execute your program with HHVM:

```
1 hhvm hello.hack
```

You should see:

```
1 Hello, World!
```

If you see this output, congratulations: you have successfully written and run your first Hack program.

Checking Types

Run the typechecker on your file:

```
hh_client
```

Since our Hello World program is correctly typed, you should see no errors. The typechecker confirms that:

- The function signature is valid
- The `echo` statement receives a string argument (which it expects)
- The return type is `void` and the function does not return any value

Now let us add a deliberate error to see what type checking looks like. Modify the program:

```
<?hh // strict

namespace MyProject;

<<__EntryPoint>>
function main(): void {
    $message: int = "Hello, World!";
    echo $message;
}
```

We have added a type annotation `: int` to `$message` but assigned it a string value. Run the typechecker:

```
hh_client
```

You will see an error message indicating that a string value cannot be assigned to an integer-typed variable. This is exactly the kind of bug that Hack's type system helps you catch before your code ever runs in production.

Running, Typing and Debugging Your First Code

Let us expand our Hello World program to demonstrate more features and show how the development workflow works end-to-end.

Create a file called `greeting.hack`:

```
<?hh // strict

namespace MyProject;

function greet(string $name): string {
    return "Hello, " . $name . "!";
}

<<__EntryPoint>>
function main(): void {
    $name = "World";
    $greeting = greet($name);
    echo $greeting . "\n";

    // Try with a different name
    $anotherGreeting = greet("Hack Developer");
    echo $anotherGreeting . "\n";
}
```

This program defines a `greet` function that takes a string parameter and returns a greeting message. The `main` function calls `greet` twice with different names.

Key observations:

- The `greet` function has typed parameters (`string $name`) and a typed return value (`: string`).
- Local variables like `$name` and `$greeting` do not have explicit type annotations because Hack infers their types from context.
- String concatenation uses the `.` operator, inherited from PHP.

Run the typechecker:

```
hh_client
```

Then run the program:

```
hhvm greeting.hack
```

Output:

```
Hello, World!
Hello, Hack Developer!
```

Intentional Error for Learning

Let us see what happens when we violate a type constraint. Modify the `main` function:

```
<<__EntryPoint>>
function main(): void {
    $name = 42;                    // Changed from string to int
    $greeting = greet($name);      // Type error: greet expects string
    echo $greeting . "\n";
}
```

Run the typechecker:

```
hh_client
```

The typechecker reports an error because `greet` expects a `string` argument but we are passing an `int`. This is a static error: it is caught before the program even runs.

Now try running the program anyway:

```
hhvm greeting.hack
```

HHVM will also catch this at runtime and throw a type error. This dual enforcement (static + runtime) is a key strength of Hack: bugs are caught both during development by the typechecker and in production by the runtime, providing defense in depth.

Essential Development Tools

A productive Hack development environment includes several tools beyond HHVM itself:

Editor Setup

Visual Studio Code: The recommended editor for Hack development is VS Code with the vscode-hack extension (developed by Slack). This extension provides:

- Syntax highlighting
- Type checking integration via `hh_client`
- Go-to-definition navigation
- Auto-completion
- Linting via HHAST

Install from the VS Code marketplace: search for "Hack" by Slack Technologies.

Vim: Use the vim-hack plugin along with ALE (Asynchronous Lint Engine) for type checking and linting integration.

Emacs: The hack-mode package provides syntax highlighting and basic integration.

Code Formatting

`hackfmt` is the official Hack code formatter, included with HHVM installations. It automatically formats your code according to consistent style rules:

```
hackfmt file.hack          ; Format in place
hackfmt --stdin            ; Format from stdin
hackfmt --diff file.hack   ; Show what would change
```

Configure your editor to run `hackfmt` on save for consistent formatting across your team.

Linting with HHAST

HHAST (Hack Abstract Syntax Tree) is a linting and code transformation tool built on Hack's full-fidelity parser. It can:

- Enforce coding style rules

- Automatically migrate code when language features change
- Perform custom static analysis

Install via Composer:

```
composer require --dev hhvm/hhast
```

Run linting:

```
vendor/bin/hhast-lint src/
```

Troubleshooting Common Installation Issues

HHVM not found after installation

If the `hhvm` command is not recognized, your PATH may not include the HHVM installation directory. On Ubuntu/Debian, HHVM is typically installed to `/usr/bin/hhvm`. Verify with:

```
which hhvm
```

If not found, check the installation logs or reinstall.

Typechecker errors on first run

When you start `hh_client` in a new project, it needs to build an initial type database. This can take a minute or two for larger projects. Be patient during the first run.

Version mismatches

Ensure that your HHVM runtime version matches the expectations of any Hack libraries you use. Check compatibility in library documentation and update dependencies as needed.

Docker permission issues

When using Docker, ensure your project directory permissions allow the container to read and write files. You may need to adjust ownership or use bind mounts carefully.

Chapter Summary

- HHVM officially supports Ubuntu and Debian on x86-64; Docker is recommended for other platforms.
- Installation options include prebuilt packages (recommended), Docker containers and building from source.
- The `.hhconfig` file configures the typechecker; `hh_client --watch` provides continuous feedback during development.
- A typical project uses Composer for dependencies and `hh_autoload.json` for autoloading configuration.
- Hack files start with `<?hh // strict`, require namespaces and use `<<__EntryPoint>>` attributes to mark entry points.
- Type annotations on function parameters and return values are enforced both statically and at runtime.
- Essential tools include `hackfmt` for formatting, HHAST for linting and editor extensions for integrated development.
- Troubleshooting common issues involves checking PATH configuration, allowing time for initial type database builds and verifying version compatibility.

In the next chapter, we will explore the foundational building blocks of Hack: syntax, variables, data types and operators.

Chapter 3: Foundations: Syntax, Variables, Data Types and Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic Syntax: Statements, Expressions and Structure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

File Structure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Statements and Expressions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Semicolons and Braces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Variables and Naming Conventions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Declaring Variables

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Naming Conventions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Local Variables and Type Inference

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Annotations on Local Variables

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Primitive Types: int, float, string, bool

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Integer (int)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Float

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

String

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Boolean (bool)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Null

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Declarations and Inference

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Function Parameter Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Return Type Declarations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Inference in Practice

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

When Inference Is Not Enough

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Constants and Readonly Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

File-Level Constants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Class Constants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Readonly Properties

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Operators: Arithmetic, Comparison, Logical and Bitwise

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Arithmetic Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Comparison Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Logical Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Bitwise Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Casting and Conversion

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Cast Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Strict Conversion Rules

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Assertions with as and ?as

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 4: Control Flow and Program Logic

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Conditional Statements: if, else, elseif

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic if Statement

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

if-else Statements

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

if-elseif-else Chains

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Nested Conditionals

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Ternary and Null Coalescing Operators

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Ternary Operator

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Null Coalescing Operator

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Switch Statements and Pattern Matching

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic switch Statement

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Fallthrough Behavior

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Multiple Values per Case

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Range Matching

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

switch as an Expression

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Loops: while, do-while, for, foreach

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

while Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

do-while Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

for Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

foreach Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Break, Continue and Goto

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

break: Exiting Loops Early

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

continue: Skipping to Next Iteration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Goto: Labeled Jumps (Use Sparingly)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Error Control Operator (@)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 5: Functions: Definition, Types and Behavior

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Defining and Calling Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic Function Definition

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Calling Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Functions Without Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Functions Without Return Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Parameters and Arguments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Typed Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Multiple Parameters with Different Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Optional Parameters and Default Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Declaring Optional Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Rules for Optional Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Multiple Optional Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Named Arguments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Named Arguments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Rules for Named Arguments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Variadic Functions and Spreading

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Variadic Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Variadic Parameters with Other Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Spreading Collections as Arguments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Lambda Expressions and Closures

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Fat Arrow Syntax

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Multiple Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Block Lambdas

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Typed Lambdas

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Closure Capture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Legacy PHP-Style Closures

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Higher-Order Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Functions as Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Returning Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Higher-Order Functions with Collections

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 6: Collections: Arrays, Vectors, Dicts, Maps, Sets and Pairs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Why Hack Replaced PHP Arrays

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

vec: Typed Indexed Collections

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What is vec?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Accessing Elements

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Safe Element Access

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Adding and Removing Elements

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Combining Vectors

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

dict: Key-Value Associative Collections

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What is dict?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Accessing Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Safe Value Access

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Merging Dictionaries

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Iterating Over Dictionaries

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

keyset: Unique Value Sets

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What is keyset?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Set Operations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

pair: Two-Element Tuples

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What is pair?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Array Syntax and Conversion

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Legacy Collections vs Hack Arrays

Converting Between Types

Collection Operations and Built-in Functions

Map: Transforming Elements

Filter: Selecting Elements

Reduce: Combining Elements

Contains and Counting

Performance Characteristics of Each Type

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 7: The Hack Type System: Static Typing, Shapes and Generics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Static vs Dynamic Typing in Practice

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Problem with Pure Dynamic Typing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Value of Static Typing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Gradual Typing: The Best of Both Worlds

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Annotations and Enforcement

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Parameter Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Return Type Annotations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Enforcement at Runtime

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Nullable Types and the ? Operator

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Declaring Nullable Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Nullable Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Nullable Parameters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The nonnull Type

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Union Types and mixed

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Union Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The mixed Type

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Narrowing with is

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Shape Types: Typed Associative Structures

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Shapes?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Optional Fields in Shapes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Open Shapes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Shape Functions from HH\Shapes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Structural Subtyping

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Tuples as Type Constructs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Tuples?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Tuple Destructuring

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Generics: Parametric Polymorphism

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Generics?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Generic Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Generic Constraints

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Generics and Collections

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 8: Object-Oriented Programming in Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Classes and Objects: Definition and Instantiation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Defining a Class

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Creating Objects

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Properties and Visibility

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Instance Properties

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Visibility Modifiers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Property Initialization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Methods and Constructors

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Instance Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Constructor Parameter Promotion

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Multiple Constructors

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Inheritance and Extending Classes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic Inheritance

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Calling Parent Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Method Overriding Rules

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Abstract Classes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Final Classes and Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Static Members and Singleton Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Static Properties and Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Singleton Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Object Comparison and Cloning

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Comparing Objects

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Cloning Objects

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 9: Interfaces, Traits, Enums and Type Organization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Interfaces: Defining Contracts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Interfaces?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Why Use Interfaces?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Interface Constants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Extending Interfaces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Implementing Multiple Interfaces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Traits: Code Reuse Without Inheritance

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Traits?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Multiple Traits

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Trait Conflicts and Aliasing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Trait Dependencies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Enumerations (enum): Typed Constants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic Enums

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

String-Based Enums

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Enum Utility Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Enums in Type Systems

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Native vs Classic Enums

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Organization Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Choosing Between Interfaces, Traits and Enums

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Combining Mechanisms

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 10: Modules, Namespaces and Project Organization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Namespaces: Scope and Organization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Namespaces?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Declaring Namespaces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Accessing Namespaced Symbols

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Sub-namespaces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Reserved Namespaces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Root Namespace

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Statements and Aliasing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Importing Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Importing with Aliases

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Importing Functions and Constants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Module System and Autoloading

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Modules?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Internal Visibility

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Modules vs Namespaces

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

File Structure Conventions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Recommended Project Layout

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Naming Conventions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

One Primary Definition Per File

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Dependencies Between Modules

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Managing Dependencies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Avoiding Circular Dependencies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Large-Scale Project Layout

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Monorepo Structure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Package Management with PACKAGES.toml

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 11: Error Handling, Exceptions and Result Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Errors vs Exceptions in Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Errors

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Exceptions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Try-Catch-Finally Blocks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Basic Try-Catch

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Multiple Catch Blocks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Finally Block

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Built-in Exception Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Creating Custom Exceptions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The AssertionError and Assertions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using assert()

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

When to Use Assertions vs Exceptions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Optional and Result Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Problem with null Returns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Optional Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Result Type Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Defensive Programming Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Fail Fast

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Guard Clauses

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 12: Asynchronous Programming and Concurrency

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Synchronous vs Asynchronous Execution

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Problem with Blocking I/O

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Asynchronous Solution

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Important: Async Is Not Multithreading

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

async Functions and Awaitables

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Declaring async Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Awaitable Type

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

await Expressions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using await Inside async Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Sequential vs Parallel with await

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Awaitable Collections and Parallelism

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Vec\from_async

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

map_async for Parallel Transformation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Dict and Keyset Async Operations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

async/Await Error Handling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Exceptions in async Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Handling Failures in Parallel Operations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Fibers and the Event Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Fibers?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Scheduler

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Concurrency Primitives

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Mutexes and Locks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Semaphores for Rate Limiting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 13: Attributes, Reflection and Metaprogramming

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Attributes (Annotations): Syntax and Usage

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Are Attributes?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Attribute Syntax

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Built-in Attributes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__Memoize

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__Pure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__Override

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__EntryPoint

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__Sealed

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__MockClass

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

__CustomVerification

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Custom Attributes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Defining Custom Attributes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using Custom Attributes for Documentation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Reflection API

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Is Reflection?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Inspecting Classes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Inspecting Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Invoking Methods Dynamically

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Runtime Introspection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Checking at Runtime

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Inspecting Enums at Runtime

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Metaprogramming Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Dependency Injection with Reflection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Generating Code from Attributes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using HHAST for Code Transformations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 14: Testing, Debugging, Profiling and Performance

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Unit Testing with HackUnit and HackTest

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Why Test in Hack?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

HackTest: The Official Testing Framework

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Test Structure and Best Practices

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

fbexpect: Expressive Assertions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Mocking and Stubbing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Debugging Techniques and Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Print Debugging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

HHVM Debugging with hphpd

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Log-Based Debugging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Logging Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Log Levels

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What to Log

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

What Not to Log

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Profiling HHVM Applications

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

XHProf Profiler

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Linux perf for HHVM

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

HHVM Built-in Profiling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Performance Optimization Techniques

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Enable JIT Compilation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Use Repo-Authoritative Mode

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Use Typed Code

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Prefer Hack Arrays Over Legacy Collections

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Minimize String Concatenation in Loops

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Batch Database Operations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Use Memoization for Expensive Operations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Profile Before Optimizing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 15: Interoperability, Migration and PHP Integration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Hack and PHP Interoperability Modes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Understanding the Relationship

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Calling PHP from Hack (Historical Context)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Calling Hack from PHP (Not Recommended)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Migrating PHP Codebases to Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Why Migrate?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Gradual Typing Strategy

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

File-by-File Migration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using the Hackificator

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Common Migration Pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Features Not Supported in Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Example: Migrating Variable Variables

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Example: Migrating Global State

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Conversion Issues

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Array Migration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Tooling for Migration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

hh_client for Incremental Checking

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

HHAST for Automated Refactoring

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Suppressing Errors During Migration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Modern Approach: New Projects in Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 16: Real-World Architecture, Design Patterns and Best Practices

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Application Architecture in Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Layered Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Example Project Structure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Dependency Flow

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Dependency Injection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Constructor Injection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Service Container

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Repository and Service Layer Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Repository Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Service Layer Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Middleware and Request Handling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Middleware Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Database Access Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Using PDO with Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Query Building

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

API Design with Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

RESTful API Structure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Response Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Input Validation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Configuration Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Environment-Based Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Coding Conventions and Style

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Naming Conventions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Code Formatting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Documentation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter 17: Security, Maintainability and Production Deployment

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Security Best Practices in Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Type Safety as a Security Feature

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Input Validation and Sanitization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Preventing Common Vulnerabilities

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

SQL Injection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Cross-Site Scripting (XSS)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Cross-Site Request Forgery (CSRF)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Path Traversal

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Authentication and Authorization Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Secure Password Handling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Role-Based Access Control

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Secure Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

HHVM Deployment Architectures

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Production Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Proxygen Integration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Docker Deployment

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Scaling Considerations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Monitoring and Observability

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Health Checks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Metrics Collection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Distributed Tracing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Long-Term Maintainability Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Code Review Practices

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Version Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Documentation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Continuous Integration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Technical Debt Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Chapter Summary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Conclusion

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Journey Complete

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

When Hack Is the Right Choice

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

The Future of Hack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

Final Thoughts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.

References

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hackprogramminglanguage.