Development Environment and Runtime - Fourward Programming Language

1. System Requirements

1.1 Hardware Requirements

- Minimum 2GB RAM
- 500MB disk space
- Any modern processor

1.2 Software Requirements

- Python 3.8 or higher
- Git (for version control)
- Text editor or IDE (e.g., VS Code, PyCharm)

---

2. Development Setup

2.1 Installation

- Clone the repository from GitHub
- Ensure Python 3.8+ is installed
- No additional dependencies required
- Run the interpreter using:

  ```
  python3 fourward_interpreter.py examples.fwd
  ```

2.2 Development Tools

- Python interpreter
- Code/text editor
- Git for source control
- Terminal for execution

---

3. Runtime Environment

3.1 Execution Requirements

- Python installed on your machine
- Access to terminal or command line interface
- No external libraries required

3.2 Environment Variables

- Not required for Fourward

---

4. Building and Running

4.1 Build Process

- No build step required
- Direct interpretation of `.fwd` source files

4.2 Execution

- Execute from terminal with:

  ```
  python3 fourward_interpreter.py <filename.fwd>
  ```

- Example:

  ```
  python3 fourward_interpreter.py examples_compatible.fwd
  ```

---

5. Testing Environment

5.1 Manual Testing

- Sample `.fwd` files included for testing
- Manual execution confirms syntax and runtime behavior

5.2 Future Testing Plans

- Integrate unit tests for tokenization, parsing, and interpretation
- Automate test case execution using Python's `unittest` or `pytest`

---

6. Debugging

6.1 Tools

- Print statements for tracing values
- Python traceback for catching errors

6.2 Techniques

- Variable inspection via print
- Step-by-step execution using logs
- Review of token streams and AST manually if needed

---

7. Deployment

7.1 Packaging

- Currently no packaging or distribution process
- Future plans may include CLI tool bundling

7.2 Distribution

- Project shared through GitHub repository
- Manual cloning and usage

---

8. Maintenance

8.1 Updates

- Versioning managed via Git commits
- Manual updates through pull and push operations

8.2 Troubleshooting

- Common issues include:
  - Unsupported syntax (e.g., unimplemented `function` blocks)
  - Typos or missing semicolons
- Use error messages and print debugging to resolve
