AIR Toolkit - Quick Start

Get started with AIR in 5 minutes.

Installation

```
# Clone repository
git clone https://github.com/LiveData-Inc/air-toolkit.git
cd air-toolkit

# Install in development mode
pip install -e .

# Verify installation
air --version
```

Create Your First Project

```
# Create a new assessment project
air init my-review --mode=review

# Change to project directory
cd my-review

# Validate project structure
air validate
```

You now have a complete AIR project:

```
my-review/
├── air-config.json      # Project configuration
├── README.md            # Project overview
├── CLAUDE.md            # AI assistant instructions
├── .air/                # Task tracking
│   ├── tasks/           # Task files go here
│   └── context/         # Project context
├── review/              # Linked resources (symlinks)
├── analysis/            # Your analysis documents
└── scripts/             # Helper scripts
```

Quick Commands

Check Status

```
air status                  # View project info
air status --format=json    # JSON output for AI
```

Validate Structure

```
air validate                # Check project structure
air validate --format=json  # JSON output
```

Task Management

```
# List tasks
air task list               # Active tasks
air task list --all         # Include archived

# Archive old tasks
air task archive --before=2025-09-01    # Archive before date
air task archive --all --dry-run        # Preview archiving
```

Common Workflows

Review Workflow (Analyze Code)

1. **Create review project:**

   ```
   air init code-review --mode=review
   cd code-review
   ```

2. **Link resources** (TODO: `air link` coming soon):

   ```
   # Manually create symlink for now
   ln -s ~/repos/target-project review/target-project
   ```

3. **Create task file** (AI assistants do this automatically):

   ```
   from datetime import datetime, timezone
   from pathlib import Path

   timestamp = datetime.now(timezone.utc).strftime("%Y%m%d-%H%M")
   task_file = f".air/tasks/{timestamp}-analyze-architecture.md"

   Path(task_file).write_text(f"""# Task: Analyze Architecture

   ## Date
   {datetime.now(timezone.utc).strftime('%Y-%m-%d %H:%M UTC')}

   ## Prompt
   Analyze the architecture of target-project

   ## Actions Taken
   1.

   ## Files Changed
   -

   ## Outcome
   ⏳ In Progress
   """)
   ```

4. **Analyze and document:**
   - Read code in `review/target-project/`

- Write analysis to `analysis/assessments/target-project.md`
- Create comparison if reviewing multiple projects

5. **Archive task when done:**

```
air task archive --all
```

Collaborate Workflow (Contribute)

1. **Create collaborative project:**

```
air init docs-improve --mode=collaborate
cd docs-improve
```

2. **Link documentation repositories:**

```
# Manually link for now
ln -s ~/repos/docs-project collaborate/docs-project
```

3. **Identify improvements:**
   - Review docs in `collaborate/docs-project/`
   - Document gaps in `analysis/improvements/`
4. **Create contributions:**
   - Place improved docs in `contributions/docs-project/`
   - Follow original structure
   - Include clear explanations
5. **Submit** (TODO: `air pr` coming soon):

```
# Manual PR for now
cd collaborate/docs-project
git checkout -b improve-docs
# Copy contributions, commit, push
gh pr create
```

AI Assistant Integration

If you're using **Claude Code** or similar AI assistants:

Check if AIR is available

```
which air
```

Use AIR commands when available

```
# Check project status
air status --format=json

# Validate structure
air validate --format=json

# List tasks
air task list --format=json
```

Task tracking is automatic

AI assistants following the CLAUDE.md instructions will:

- Create task files for every piece of work
- Update with outcomes
- Archive when appropriate

Project Modes

Review Mode

```
air init project --mode=review
```

- **Purpose**: Analyze codebases (READ-ONLY)
- **Directories**: `review/` , `analysis/assessments/`
- **Use case**: Compare implementations, identify patterns

Collaborate Mode

```
air init project --mode=collaborate
```

- **Purpose**: Contribute improvements
- **Directories**: `collaborate/`, `contributions/`
- **Use case**: Documentation improvements, code contributions

Mixed Mode (Default)

```
air init project --mode=mixed
# or simply:
air init project
```

- **Purpose**: Both review and collaborate
- **Directories**: All of the above
- **Use case**: Complex assessments with some contributions

Task Archiving

Keep `.air/tasks/` organized:

```
# Archive tasks older than 30 days
air task archive --before=2025-09-01

# Archive all tasks (preview first)
air task archive --all --dry-run
air task archive --all

# View archive statistics
air task archive-status

# Restore if needed
air task restore 20251003-1430
```

Archived tasks move to `.air/tasks/archive/YYYY-MM/` and remain accessible with `air task list --all`.

JSON Output

All status commands support `--format=json` for AI parsing:

```
air status --format=json
air validate --format=json
air task list --format=json
air task archive-status --format=json
```

Next Steps

- **Read full docs**: See `docs/` directory
  - `COMMANDS.md` - Complete command reference
  - `SPECIFICATION.md` - Feature specifications
  - `AI-INTEGRATION.md` - AI assistant integration guide
  - `TASK-ARCHIVE-DESIGN.md` - Task archiving details
- **Example project**: Try the workflows above with a real repository
- **AI integration**: Use with Claude Code or similar AI assistants
- **Contribute**: See `DEVELOPMENT.md` for contributing guidelines

Getting Help

```
# Command help
air --help
air init --help
air task --help

# Documentation
cat docs/COMMANDS.md

# Report issues
https://github.com/LiveData-Inc/air-toolkit/issues
```

Quick Reference

Command	Description
`air init <name>`	Create new project
`air validate`	Validate project structure

Command	Description
`air status`	Show project status
`air task list`	List active tasks
`air task list --all`	Include archived tasks
`air task archive --all`	Archive all tasks
`air task restore <id>`	Restore archived task
`air task archive-status`	Show archive stats

Ready to dive deeper? Check out the full documentation or try creating your first assessment project!