IOblend

DEVELOPER EDITION / INSTALLATION

Install IOblend on macOS

From download to the first working local pipeline.

A practical installation guide

Set up Designer and the local Engine, configure your development environment, and build your first validated pipeline.



NOVEMBER 2026 / DEVELOPER EDITION

01 / PREPARE

Start with Developer Edition

Designer and Engine run on the Mac for development and testing. Remote production Engine deployment uses Enterprise Edition.

What you will install

The IOblend Mac package, a tested Python environment with supporting modules, the local Engine and Designer, then your first project Run Parameters.

Requirement	Before you begin
Mac workstation	A modern Mac; approximately 16 GB RAM recommended
Local storage	Allow around 30 GB free SSD space where practical; software download is approximately 2 GB
Internet	Download software, Python and dependencies; reach data systems only with authorised access
Python	3.10.11 for this documented Developer Edition package
Terminal and shell	Terminal access and ability to update the active shell profile
Archive extractor	7-Zip-compatible utility; example uses the 7zz command

Release compatibility

These steps follow the currently documented Mac package and its Python 3.10.11 dependency. Confirm any change of Python version against the IOblend package before upgrading the local runtime.

02 / PYTHON SETUP

Install and isolate Python

Use a dedicated virtual environment so the required packages do not interfere with unrelated Python projects.

01

Install the documented Python version

Obtain Python 3.10.11 from Python.org using the macOS 64-bit universal installer. Verify the interpreter used by Terminal.

TERMINAL - VERIFY PYTHON

```
python3 --version
python3 -c "import os,sys; print(os.path.dirname(sys.executable))"
```

02

Create a dedicated virtual environment

Create a local venv and activate it before installing dependencies. The example uses a folder under your home directory.

TERMINAL - CREATE AND ACTIVATE VENV

```
mkdir -p "$HOME/python_venvs"
python3 -m venv "$HOME/python_venvs/p310"
source "$HOME/python_venvs/p310/bin/activate"
python3 --version
```

Confirm the interpreter

The activated environment should report Python 3.10.11. If it does not, check which Python installation created the venv before continuing.

03 / PYTHON MODULES

Install the required dependency set

Run these commands only after activating the IOblend virtual environment.

The original Mac installation specifies the following packages. Keeping them together makes the procedure easier to repeat and review.

TERMINAL - INSTALL SUPPORTING MODULES

```
python3 -m pip install pyinstaller Pillow Tornado psutil \
  boto3 retrying tzlocal py4j tabulate pandas cryptography

python3 -m pip install --upgrade google-cloud-storage

python3 -m pip install azure-storage-blob azure-identity \
  azure-common azure-storage-file-datalake pyarmor
```

Keep the environment active

Remain inside the virtual environment while preparing to launch IOblend. Use `deactivate` when you finish working with this environment.

If a package fails to install, capture the first error and confirm both the interpreter version and the active virtual environment. Do not replace the tested Python version solely to resolve an unrelated package error.

Python lifecycle note

Python 3.10.11 is upstream end-of-life as of November 2026. The IOblend team should validate a replacement runtime before this documented dependency is changed.

04 / EXTRACT AND LAUNCH

Extract the complete IOblend package

Use a 7-Zip-based tool on macOS, then launch the application from the extracted directory.

01

Download the macOS distribution

Complete the Developer Edition download form, choose macOS and save the complete ioblend_designer_mac.zip locally.

02

Test and extract with 7zz

Use a 7-Zip-based tool rather than the default macOS archive utility. Adapt the filename or download location if yours differs.

TERMINAL - EXAMPLE ARCHIVE EXTRACTION

```
mkdir -p "$HOME/IOblend"
7zz t "$HOME/Downloads/ioblend_designer_mac.zip"
7zz x "$HOME/Downloads/ioblend_designer_mac.zip" \
  -o"$HOME/IOblend"
```

03

Open the Designer executable

Navigate to the extracted ioblend_designer folder and launch the executable. Keep the terminal and Playbook Designer windows open.

Expected first launch

The full distribution should be present. Both Designer and the Script/terminal window should open without missing-module or incomplete-archive errors.

05 / ENVIRONMENT

Configure shell environment variables

Designer generates the required environment export statements; apply them to the shell profile used by your workstation.

01

Generate the environment setup file

In Designer choose Setup > Setup Environmental Vars. The application creates io_env_setup.sh inside the IOblend Designer directory.

02

Review and add the export statements

Inspect io_env_setup.sh, then copy the required export statements into the appropriate active shell profile, commonly ~/.zprofile on zsh systems.

TERMINAL - INSPECT AND REFRESH PROFILE

```
cat /path/to/ioblend_designer/io_env_setup.sh

# After editing the appropriate shell profile:
source "$HOME/.zprofile"
```

03

Restart the application

Close both Designer and its terminal window, then reopen ioblend_designer so the new environment is inherited.

Where to look if setup fails

Check the correct shell profile was edited, export statements were copied accurately, and the required folders exist in the extracted package.

06 / MACOS FIRST LAUNCH

Use the system-approved app opening method

macOS may block the first launch of an application from an unidentified developer.

Review the application first

Confirm the IOblend package came from the official download flow. If macOS blocks the app, use Finder and the approved app-specific Open / Open Anyway process in Privacy & Security, where offered.

The earlier installation guide described disabling Gatekeeper globally. That is unnecessary for normal first-launch approval and should not be the standard procedure. Follow any organisation-managed workstation restrictions.

Symptom	First check
App cannot be opened	Verify downloaded package; use the app-specific macOS security approval process.
Designer opens; terminal closes	Confirm activated Python environment, required packages and shell profile exports.
Module not found	Check the active venv and installed package list.
Archive files missing	Re-extract with the 7-Zip-based tool.
Java/Spark connection failure	Check Script output, runtime resources and endpoint controls.

Keep useful diagnostics

Record macOS version, Python version, Engine version, the first terminal error and the result of the environment setup. Remove credentials or sensitive data before sharing logs.

07 / PROJECT SETUP

Create the project and Run Parameters

Complete the local runtime configuration before adding Source, Transformation and Sink components.

SUGGESTED PROJECT DIRECTORIES

```
$HOME/IOblendLabs/first-pipeline/
  config/          playbooks/        transforms/
  source_data/     output/           checkpoint/
  state/           interaction/      jars/
```

Keep source and sink directories separate. Avoid reusing checkpoints across unrelated experiments. Keep credentials outside shared repositories and test with isolated input data.

In Designer select **Config > Project Run Params > New Run Params**. Save the project configuration with the local paths below.

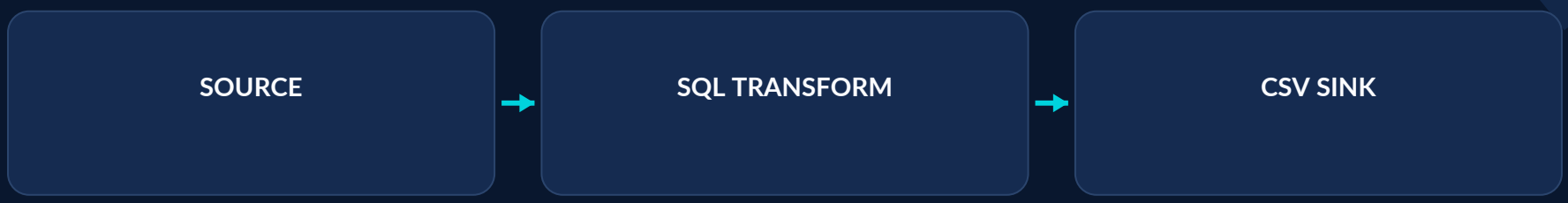
Setting	Local first-run value
Pipeline Location	/playbooks
PipeLine Name	first_pipeline_01
Transform Ext / State Directory	/transforms / /state
File System / Spark Master / SSH	local / local / local
Epochs	1 (0 for continuous mode)
Debug	true
Run Interactive	true
interaction_by_file	true

Before building: Reopen the saved Run Parameters and confirm the pipeline name, project paths, and three development-time debugging options.

08 / FIRST PIPELINE

Make the first result correct

Add each component, run it, inspect its DataFrame, then add the next one.



01

Read

Start with a small local CSV. Confirm expected headers and source record counts.

02

Transform

Apply a simple SQL calculation, verify its output and inspect the intermediate DataFrame.

03

Write

Save to a separate output folder. Verify actual rows, calculations and generated metadata.

If the Engine does not start

Check the Python virtual environment, installed modules, shell profile, extracted files, and terminal errors before changing your pipeline definitions. The Developer Handbook provides detailed Spark and project troubleshooting.

09 / READY TO BUILD

Installation complete. Build something.

The next step is a small, working pipeline - built one component at a time.

The best first experiment is deliberately simple: connect a local CSV source, add a SQL transformation and write the result to a separate destination. Inspect each stage before continuing.

Your first success criteria

Designer and Script/terminal windows open correctly; project Run Parameters are saved; a Source returns the expected rows; a SQL transformation produces verified results; and the Sink writes the expected output.

CONTINUE THE LEARNING PATH

01 / Developer Handbook

Use the guided CSV lab, Run Parameters reference and troubleshooting sections.

02 / Official video tutorials

Move from the first source through transformations, sinks and streaming.

03 / Enterprise Edition

When your pipeline is ready for production, discuss remote deployment and operations.

DOWNLOAD DEVELOPER EDITION

DOCUMENTATION & VIDEOS

ioblend.com/download/ | ioblend.com/documentation/
Enterprise deployment? ioblend.com/contact/