Installation using Isaac Sim Pip Package

Contents

- Installing Isaac Sim
- Installing Isaac Lab

The following steps first installs Isaac Sim from pip, then Isaac Lab from source code.

Attention

Installing Isaac Sim with pip requires GLIBC 2.35+ version compatibility. To check the GLIBC version on your system, use command `ldd --version`.

This may pose compatibility issues with some Linux distributions. For instance, Ubuntu 20.04 LTS has GLIBC 2.31 by default. If you encounter compatibility issues, we recommend following the Isaac Sim Binaries Installation approach.

Note

If you plan to Set up Visual Studio Code later, we recommend following the Isaac Sim Binaries Installation approach.

Installing Isaac Sim

From Isaac Sim 4.0 onwards, it is possible to install Isaac Sim using pip. This approach makes it easier to install Isaac Sim without requiring to download the Isaac Sim binaries. If you encounter any issues, please report them to the Isaac Sim Forums.

Attention

On Windows, it may be necessary to enable long path support to avoid installation errors due to OS limitations.

Preparing a Python Environment

Creating a dedicated Python environment is **strongly recommended**. It helps:

- **Avoid conflicts with system Python** or other projects installed on your machine.
- **Keep dependencies isolated**, so that package upgrades or experiments in other projects do not break Isaac Sim.
- **Easily manage multiple environments** for setups with different versions of dependencies.
- **Simplify reproducibility** — the environment contains only the packages needed for the current project, making it easier to share setups with colleagues or run on different machines.

You can choose different package managers to create a virtual environment.

- **UV**: A modern, fast, and secure package manager for Python.
- **Conda**: A cross-platform, language-agnostic package manager for Python.
- **venv**: The standard library for creating virtual environments in Python.

Caution

The Python version of the virtual environment must match the Python version of Isaac Sim.

- For Isaac Sim 5.X, the required Python version is 3.11.
- For Isaac Sim 4.X, the required Python version is 3.10.

Using a different Python version will result in errors when running Isaac Lab.

The following instructions are for Isaac Sim 5.X, which requires Python 3.11. If you wish to install Isaac Sim 4.5, please use modify the instructions accordingly to use Python 3.10.

- Create a virtual environment using one of the package managers:

UV Environment | Conda Environment | venv Environment

To install `uv`, please follow the instructions here.

> **Note**
>
> A virtual environment created by `uv venv` does **not** include `pip`. Since Isaac Lab installation requires `pip`, please install it manually after activating the environment.

You can create the Isaac Lab environment using the following commands:

Linux | Windows

```
# create a virtual environment named env_isaaclab with python3.11 and pip
uv venv --python 3.11 --seed env_isaaclab
# activate the virtual environment
source env_isaaclab/bin/activate
```

- Ensure the latest pip version is installed. To update pip, run the following command from inside the virtual environment:

Linux | Windows

```
pip install --upgrade pip
```

Installing dependencies

> **Note**
>
> In case you used UV to create your virtual environment, please replace `pip` with `uv pip` in the following commands.

- Install Isaac Sim pip packages:

```
pip install "isaacsim[all,extscache]==5.1.0" --extra-index-url https://pypi.nvi
```

- Install a CUDA-enabled PyTorch build that matches your system architecture:

Linux (x86_64) | Windows (x86_64) | Linux (aarch64)

```
pip install -U torch==2.7.0 torchvision==0.22.0 --index-url https://download
```

Verifying the Isaac Sim installation

- Make sure that your virtual environment is activated (if applicable)
- Check that the simulator runs as expected:

```
# note: you can pass the argument "--help" to see all arguments possible.
isaacsim
```

- It's also possible to run with a specific experience file, run:

```
# experience files can be absolute path, or relative path searched in isaacsim/a
isaacsim isaacsim.exp.full.kit
```

Note

When running Isaac Sim for the first time, all dependent extensions will be pulled from the registry. This process can take upwards of 10 minutes and is required on the first run of each experience file. Once the extensions are pulled, consecutive runs using the same experience file will use the cached extensions.

Attention

The first run will prompt users to accept the Nvidia Omniverse License Agreement. To accept the EULA, reply `Yes` when prompted with the below message:

```
By installing or using Isaac Sim, I agree to the terms of NVIDIA OMNIVERSE
in https://docs.isaacsim.omniverse.nvidia.com/latest/common/NVIDIA_Omnivers

Do you accept the EULA? (Yes/No): Yes
```

If the simulator does not run or crashes while following the above instructions, it means that something is incorrectly configured. To debug and troubleshoot, please check Isaac Sim

documentation and the Isaac Sim Forums.

Installing Isaac Lab

Cloning Isaac Lab

> **Note**
>
> We recommend making a fork of the Isaac Lab repository to contribute to the project but this is not mandatory to use the framework. If you make a fork, please replace `isaac-sim` with your username in the following instructions.

Clone the Isaac Lab repository into your project's workspace:

SSH | HTTPS

```
git clone git@github.com:isaac-sim/IsaacLab.git
```

We provide a helper executable isaaclab.sh and isaaclab.bat for Linux and Windows respectively that provides utilities to manage extensions.

Linux | Windows

```
./isaaclab.sh --help

usage: isaaclab.sh [-h] [-i] [-f] [-p] [-s] [-t] [-o] [-v] [-d] [-n] [-c] -- Util

optional arguments:
   -h, --help           Display the help content.
   -i, --install [LIB]  Install the extensions inside Isaac Lab and learning fram
   -f, --format         Run pre-commit to format the code and check lints.
   -p, --python         Run the python executable provided by Isaac Sim or virtua
   -s, --sim            Run the simulator executable (isaac-sim.sh) provided by I
   -t, --test           Run all python pytest tests.
   -o, --docker         Run the docker container helper script (docker/container
   -v, --vscode         Generate the VSCode settings file from template.
   -d, --docs           Build the documentation from source using sphinx.
   -n, --new            Create a new external project or internal task from templ
   -c, --conda [NAME]   Create the conda environment for Isaac Lab. Default name
   -u, --uv [NAME]      Create the uv environment for Isaac Lab. Default name is
```

Installation

- Install dependencies using `apt` (on Linux only):

```
# these dependency are needed by robomimic which is not available on Windows
sudo apt install cmake build-essential
```

- Run the install command that iterates over all the extensions in `source` directory and installs them using pip (with `--editable` flag):

Linux | Windows

```
./isaaclab.sh --install # or "./isaaclab.sh -i"
```

By default, the above will install **all** the learning frameworks. These include `rl_games`, `rsl_rl`, `sb3`, `skrl`, `robomimic`.

If you want to install only a specific framework, you can pass the name of the framework as an argument. For example, to install only the `rl_games` framework, you can run:

Linux | Windows

```
./isaaclab.sh --install rl_games  # or "./isaaclab.sh -i rl_games"
```

The valid options are `all`, `rl_games`, `rsl_rl`, `sb3`, `skrl`, `robomimic`, and `none`. If `none` is passed, then no learning frameworks will be installed.

Verifying the Isaac Lab installation

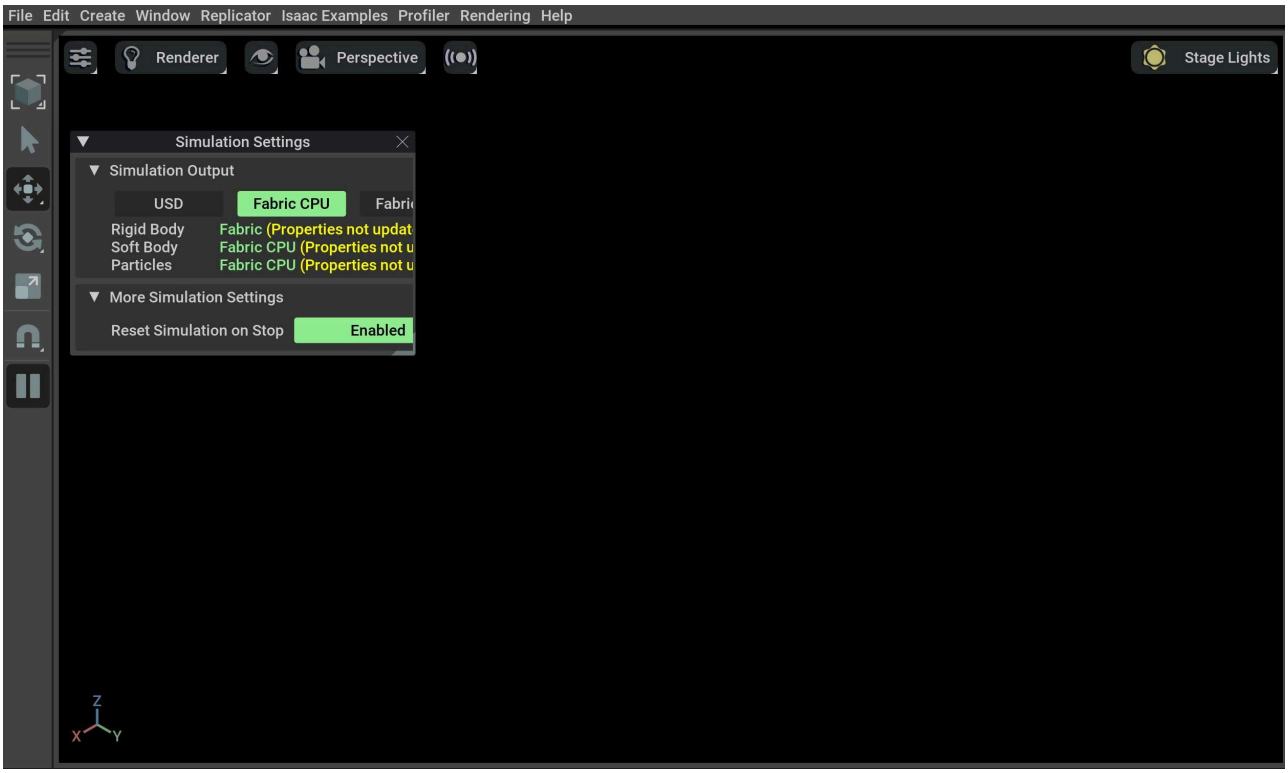
To verify that the installation was successful, run the following command from the top of the repository:

Linux | Windows

```
# Option 1: Using the isaaclab.sh executable
# note: this works for both the bundled python and the virtual environment
./isaaclab.sh -p scripts/tutorials/00_sim/create_empty.py

# Option 2: Using python in your virtual environment
python scripts/tutorials/00_sim/create_empty.py
```

The above command should launch the simulator and display a window with a black viewport. You can exit the script by pressing `Ctrl+C` on your terminal. On Windows machines, please terminate the process from Command Prompt using `Ctrl+Break` or `Ctrl+fn+B`.



If you see this, then the installation was successful! 🎉

Note

If you see an error `ModuleNotFoundError: No module named 'isaacsim'`, please ensure that the virtual environment is activated and `source _isaac_sim/setup_conda_env.sh` has been executed (for uv as well).

Train a robot!

You can now use Isaac Lab to train a robot through Reinforcement Learning! The quickest way to use Isaac Lab is through the predefined workflows using one of our **Batteries-**

included robot tasks. Execute the following command to quickly train an ant to walk! We recommend adding `--headless` for faster training.

Linux | Windows

```
./isaaclab.sh -p scripts/reinforcement_learning/rsl_rl/train.py --task=Isaac-Ant
```

... Or a robot dog!

Linux | Windows

```
./isaaclab.sh -p scripts/reinforcement_learning/rsl_rl/train.py --task=Isaac-Velo
```

Isaac Lab provides the tools you'll need to create your own **Tasks** and **Workflows** for whatever your project needs may be. Take a look at our How-to Guides guides like Adding your own learning Library or Wrapping Environments for details.