Getting started with Anaconda

Anaconda is a powerful package manager and environment manager that you use with command line commands at the Anaconda Prompt for Windows, or in a Terminal window for macOS or Linux.

This 20-minute guide to getting started with Anaconda lets you try out the major features of conda. You should understand how Anaconda works when you finish this guide.

Before you start

You should have already installed Anaconda.

Installing on Windows

1. Download the installer:
    - Miniconda installer for Windows.
    - Anaconda installer for Windows.
2. Double-click the `.exe` file.
3. Follow the instructions on the screen.

    If you are unsure about any setting, accept the defaults. You can change them later.

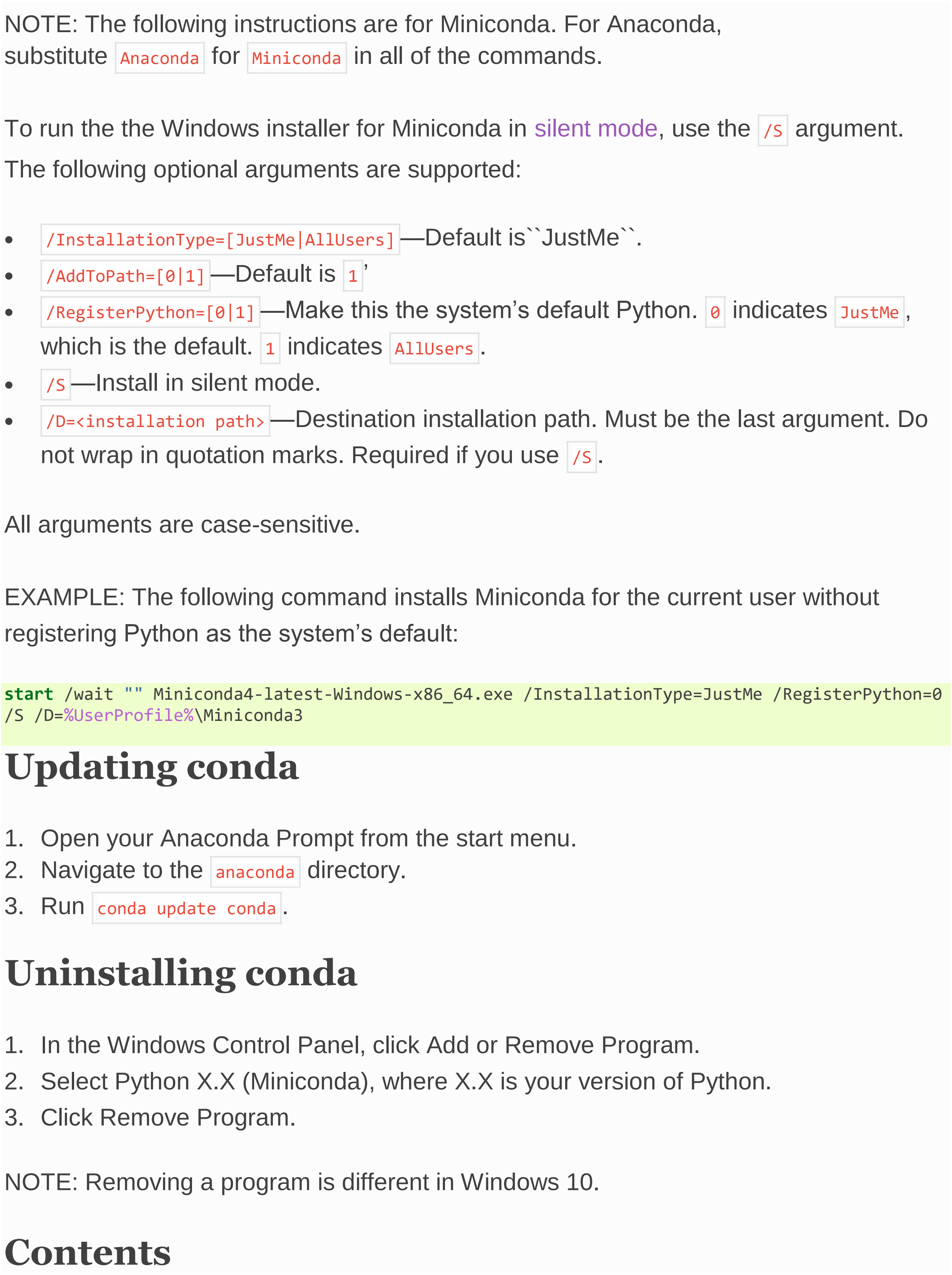
    When installation is finished, from the **Start** menu, open the Anaconda Prompt.

Testing your installation

To test your installation, in your Terminal window or Anaconda Prompt, run the command `conda list`.

For a successful installation, a list of installed packages appears.

Installing in silent mode



NOTE: The following instructions are for Miniconda. For Anaconda, substitute `Anaconda` for `Miniconda` in all of the commands.

To run the the Windows installer for Miniconda in silent mode, use the `/S` argument. The following optional arguments are supported:

- `/InstallationType=[JustMe|AllUsers]`—Default is``JustMe``.
- `/AddToPath=[0|1]`—Default is `1`’
- `/RegisterPython=[0|1]`—Make this the system’s default Python. `0` indicates `JustMe`, which is the default. `1` indicates `AllUsers`.
- `/S`—Install in silent mode.
- `/D=<installation path>`—Destination installation path. Must be the last argument. Do not wrap in quotation marks. Required if you use `/S`.

All arguments are case-sensitive.

EXAMPLE: The following command installs Miniconda for the current user without registering Python as the system’s default:

```
start /wait "" Miniconda4-latest-Windows-x86_64.exe /InstallationType=JustMe /RegisterPython=0
/S /D=%UserProfile%\Miniconda3
```

Updating conda

1. Open your Anaconda Prompt from the start menu.
2. Navigate to the `anaconda` directory.
3. Run `conda update conda`.

Uninstalling conda

1. In the Windows Control Panel, click Add or Remove Program.
2. Select Python X.X (Miniconda), where X.X is your version of Python.
3. Click Remove Program.

NOTE: Removing a program is different in Windows 10.

Contents

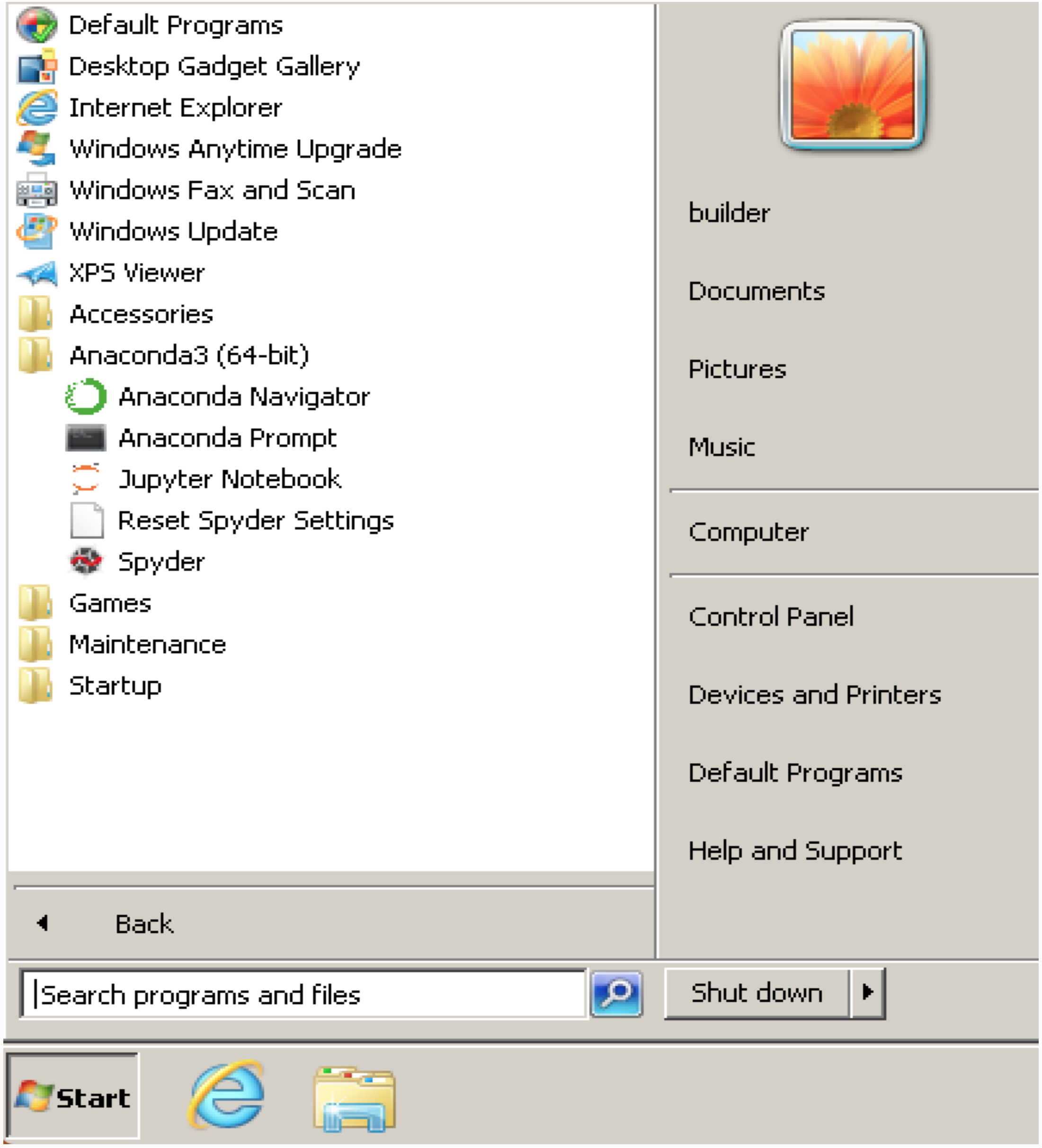
- Starting conda on Windows, macOS or Linux. 2 MINUTES
- Managing conda. Verify that Anaconda is installed and check that conda is updated to the current version. 3 MINUTES
- Managing environments. Create environments and move easily between them. 5 MINUTES
- Managing Python. Create an environment that has a different version of Python. 5 MINUTES
- Managing packages. Find packages available for you to install. Install packages. 5 MINUTES

TOTAL TIME: 20 MINUTES

Starting conda

Windows

- From the Start menu, search for and open “Anaconda Prompt”.



On Windows, all commands below are typed into the Anaconda Prompt window.

MacOS

- Open Launchpad, then click the Terminal icon.

On macOS, all commands below are typed into the Terminal window.

Linux

- Open a Terminal window.

On Linux, all commands below are typed into the Terminal window.

Managing conda

Verify that conda is installed and running on your system by typing:

```
conda --version
```

Conda displays the number of the version that you have installed. You do not need to navigate to the Anaconda directory.

EXAMPLE: `conda 4.4.9`

NOTE: If you get an error message, make sure you closed and re-opened the Terminal window after installing, or do it now. Then verify that you are logged into the same user account that you used to install Anaconda or Miniconda.

Update conda to the current version. Type the following:

```
conda update conda
```

Conda compares versions and then displays what is available to install.

If a newer version of conda is available, type `y` to update:

```
Proceed ([y]/n)? y
```

TIP: We recommend that you always keep conda updated to the latest version.

Managing Environments

Conda allows you to to create separate environments containing files, packages and their dependencies that will not interact with other environments.

When you begin using Anaconda, you already have a default environment named `base`. You don’t want to put programs into your base environment, though. Create separate environments to keep your programs isolated from each other.

1. Create a new environment and install a package in it.

   We will name the environment `snowflakes` and install the package BioPython. At the Anaconda Prompt or in your Terminal window, type the following:

   ```
   conda create --name snowflakes biopython
   ```

   Conda checks to see what additional packages (“dependencies”) Biopython will need, and asks if you want to proceed:

   ```
   Proceed ([y]/n)? y
   ```

   Type “y” and press Enter to proceed.
2. To use, or “activate” the new environment, type the following:

   - Windows: `activate snowflakes`
   - Linux and macOS: `source activate snowflakes`

   Now that you are in your `snowflakes` environment, any conda commands you type will go to that environment until you deactivate it.
3. To see a list of all your environments, type:

   ```
   conda info --envs
   ```

   A list of environments appears, similar to the following:

   ```
   conda environments:

       base           /home/username/Anaconda3
       snowflakes   * /home/username/Anaconda3/envs/snowflakes
   ```

   TIP: The active environment is the one with an asterisk (*).
4. Change your current environment back to the default (base):

   - Windows: `deactivate`
   - Linux, macOS: `source deactivate`

   TIP: When the environment is deactivated, its name is no longer shown in your prompt, and the asterisk (*) returns to base. To verify, you can repeat the `conda info --envs` command.

Managing Python

When you create a new environment, conda installs the same Python version you used when you downloaded and installed Anaconda. If you want to use a different version of Python, for example Python 3.5, simply create a new environment and specify the version of Python that you want.

1. Create a new environment named “snakes” that contains Python 3.5:

   ```
   conda create --name snakes python=3.5
   ```

   When conda asks if you want to proceed, type “y” and press Enter.
2. Activate the new environment:
   - Windows: `activate snakes`
   - Linux, macOS: `source activate snakes`
3. Verify that the snakes environment has been added and is active:

   ```
   conda info --envs
   ```

   Conda displays the list of all environments with an asterisk (*) after the name of the active environment:

   ```
   # conda environments:
   #
   base                      /home/username/anaconda3
   snakes                 *  /home/username/anaconda3/envs/snakes
   snowflakes                /home/username/anaconda3/envs/snowflakes
   ```

   The active environment is also displayed in front of your prompt in (parentheses) or [brackets] like this:

   ```
   (snakes) $
   ```

4. Verify which version of Python is in your current environment:

   ```
   python --version
   ```

5. Deactivate the snakes environment and return to base environment:
   - Windows: `deactivate`
   - Linux, macOS: `source deactivate`

Managing packages

In this section, you check which packages you have installed, check which are available and look for a specific package and install it.

1. To find a package you have already installed, first activate the environment you want to search. Look above for the commands to activate your snakes environment.
2. Check to see if a package you have not installed named “beautifulsoup4” is available from the Anaconda repository (must be connected to the Internet):

   ```
   conda search beautifulsoup4
   ```

   Conda displays a list of all packages with that name on the Anaconda repository, so we know it is available.
3. Install this package into the current environment:

   ```
   conda install beautifulsoup4
   ```

4. Check to see if the newly installed program is in this environment:

   ```
   conda list
   ```