

## What is NetworkX?

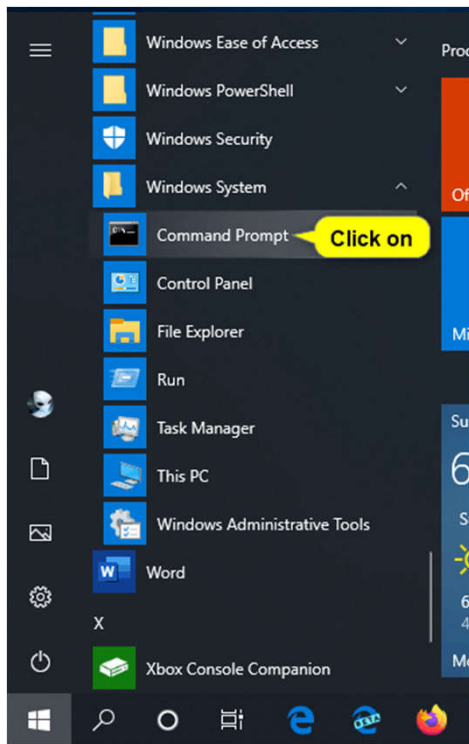
A Python package for the creation and manipulation of complex networks.

- Data structures for graphs, digraphs, and multigraphs
- Many standard graph algorithms
- Network structure and analysis measures
- Generators for classic graphs, random graphs, and synthetic networks

## How to install NetworkX

<https://networkx.github.io/documentation/stable/install.html>

- NetworkX requires Python 3.5, 3.6, 3.7, or 3.8.
- You should type ***python --version*** in **command prompt(power shell or 命令提示符 or terminal)** to check the version of your python before installing the package.
- Use command ***pip install networkx*** to install NetworkX.
- Use command ***conda install -c anaconda network*** to install NetworkX if you are using conda environment



```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [版本 10.0.16299.251]
(c) 2017 Microsoft Corporation。保留所有权利。

C:\Users\Admin>python --version
Python 3.6.3 :: Anaconda custom (64-bit)

C:\Users\Admin>python
Python 3.6.3 |Anaconda custom (64-bit)| (default, Oct 15 2017, 03:27:45) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

## Create a graph

Import the networks package

```
>>> import networkx as nx
```

Create an empty graph with no nodes and no edges.

```
>>> G = nx.Graph()
```

Create an empty directed graph.

```
>>> G = nx.DiGraph()
```

Add one or more nodes to the graph G.

```
>>> G.add_node(node)
```

```
>>> G.add_nodes_from([node1, node2...])
```

Add one or more edges to the graph G.

```
>>> G.add_edge(u, v)
```

```
>>> G.add_edges_from([(u1, v1), (u2, v2)...])
```

<https://networkx.github.io/documentation/stable/tutorial.html#creating-a-graph>

## Algorithms

Build dfs tree from source node.

```
>>> T = nx.dfs_tree(graph, source_node)
```

Print the traversing process of dfs from the source node

```
>>> T.edges()
```

<https://networkx.github.io/documentation/stable/reference/algorithms/traversal.html?highlight=bfs%23>

Print the shorts path from source node to target node using Dijkstra algorithm

```
>>> dijkstra_path(G, source, target, weight='weight')
```

[https://networkx.github.io/documentation/stable/reference/algorithms/shortest\\_paths.html](https://networkx.github.io/documentation/stable/reference/algorithms/shortest_paths.html)

## Online document

The screenshot shows the NetworkX 2.4 documentation website. A sidebar on the left contains a search bar and a list of links: Install, Tutorial, Reference, Developer Guide, Release Log, License, Credits, Citing, Bibliography, and Examples. The main content area displays the 'Overview of NetworkX' page. Red callouts highlight the search bar with the text 'Search everything about NetworkX here', the 'Install' link with 'Install guide', and the 'Tutorial' link with 'quick start'.

Search everything about NetworkX here

Install guide

quick start

NetworkX 2.4

Search docs

Install

Tutorial

Reference

Developer Guide

Release Log

License

Credits

Citing

Bibliography

Examples

Docs » Overview of NetworkX

### Overview of NetworkX

NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

NetworkX provides:

- tools for the study of the structure and dynamics of social, biological, and infrastructure networks;
- a standard programming interface and graph implementation that is suitable for many applications;
- a rapid development environment for collaborative, multidisciplinary projects;
- an interface to existing numerical algorithms and code written in C, C++, and FORTRAN; and
- the ability to painlessly work with large nonstandard data sets.

With NetworkX you can load and store networks in standard and nonstandard data formats, generate many types of random and classic networks, analyze network structure, build network

<https://networkx.github.io/documentation/stable/index.html>