Pytz: Quick Reference Guide

This guide covers installation, key functionalities, and code examples for the pytz library.

What is pytz?

Pytz is a Python library for timezone conversion.

Installation

Installation using pip:

```
pip install pytz
```

Importation

After installing, import into your code using:

```
import pytz
```

Utility Methods

`pytz.all_timezones` Returns a list of all pytz supported time zones.[1]

`pytz.common_timezones` Returns a list of the most common time zones.

`pytz.country_timezones` Returns a dictionary that maps country codes to the supported timezones in each country.

`pytz.utc` Returns a `tzinfo` object representing UTC[2].

Terminology & Resources

1. Olson Time Zone Database
2. UTC: Coordinated Universal Time
3. DST: Daylight Savings Time
4. GitHub

Core Methods

`timezone(zone: str)` Creates a `timezone` object given a timezone name.

```
eastern = pytz.timezone('US/Eastern)
```

`tzname(dt: datetime)` Returns the time zone name of a `datetime` object.

```
name = eastern.tzname(localized_dt)
```

`localize(dt: datetime)` Localizes a naive `datetime` object to a specific timezone.

```
eastern = pytz.timezone('US/Eastern')
naive_dt = datetime(2022, 4, 30, 12)
local_dt = eastern.localize(naive_dt)
```

`normalize(dt: datetime)` Normalizes a localized `datetime` object, adjusting it for DST[3].

```
normalized_dt = local_dt.normalize()
```

`dst(dt: datetime)` Returns the DST[3] offset (`timedelta` object) for the given `datetime` object.

```
tz = pytz.timezone('America/New_York')
date_in_dst = datetime(2024, 4, 30)
is_dst = tz.dst(date_in_dst)
```

`utcoffset(dt: datetime)` Returns the UTC[2] offset of a `datetime` object in a specific timezone.

```
offset = localized_dt.utcoffset()
```