Financial Data Analysis with Python

Instructor: Luping Yu

Apr 26, 2022

Practice 01. Get Data - Application Programming Interface

Yahoo Finance

yfinance offers a threaded and Pythonic way to download market data from Yahoo! finance

yfinance is not affiliated, endorsed, or vetted by Yahoo, Inc. It's an open-source tool that uses Yahoo's publicly available **APIs**, and is intended for research and educational purposes.

Install **yfinance** using pip: $ **pip install yfinance**

Introducing the *Ticker( )* module:

In [1]:
```
import yfinance as yf

msft = yf.Ticker("MSFT")
print(msft)
```

```
yfinance.Ticker object <MSFT>
```

In [2]:
```
# get stock info
msft.info
```

Out[2]:

```
{'zip': '98052-6399',
 'sector': 'Technology',
 'fullTimeEmployees': 181000,
 'longBusinessSummary': 'Microsoft Corporation develops, licenses, and suppo
rts software, services, devices, and solutions worldwide. Its Productivity a
nd Business Processes segment offers Office, Exchange, SharePoint, Microsoft
Teams, Office 365 Security and Compliance, and Skype for Business, as well a
s related Client Access Licenses (CAL); Skype, Outlook.com, OneDrive, and Li
nkedIn; and Dynamics 365, a set of cloud-based and on-premises business solu
tions for organizations and enterprise divisions. Its Intelligent Cloud segm
ent licenses SQL, Windows Servers, Visual Studio, System Center, and related
CALs; GitHub that provides a collaboration platform and code hosting service
for developers; and Azure, a cloud platform. It also offers support services
and Microsoft consulting services to assist customers in developing, deployi
ng, and managing Microsoft server and desktop solutions; and training and ce
rtification on Microsoft products. Its More Personal Computing segment provi
des Windows original equipment manufacturer (OEM) licensing and other non-vo
lume licensing of the Windows operating system; Windows Commercial, such as
volume licensing of the Windows operating system, Windows cloud services, an
d other Windows commercial offerings; patent licensing; Windows Internet of
Things; and MSN advertising. It also offers Surface, PC accessories, PCs, ta
blets, gaming and entertainment consoles, and other devices; Gaming, includi
ng Xbox hardware, and Xbox content and services; video games and third-party
video game royalties; and Search, including Bing and Microsoft advertising.
It sells its products through OEMs, distributors, and resellers; and directl
y through digital marketplaces, online stores, and retail stores. It has col
laborations with Dynatrace, Inc., Morgan Stanley, Micro Focus, WPP plc, ACI
Worldwide, Inc., and iCIMS, Inc., as well as strategic relationships with Av
aya Holdings Corp. and wejo Limited. Microsoft Corporation was founded in 19
75 and is based in Redmond, Washington.',
 'city': 'Redmond',
 'phone': '425 882 8080',
 'state': 'WA',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.microsoft.com',
 'maxAge': 1,
 'address1': 'One Microsoft Way',
 'industry': 'Software—Infrastructure',
 'ebitdaMargins': 0.49123,
 'profitMargins': 0.38498002,
 'grossMargins': 0.68825996,
 'operatingCashflow': 83909001216,
 'revenueGrowth': 0.201,
 'operatingMargins': 0.42523998,
 'ebitda': 90829996032,
 'targetLowPrice': 306.55,
 'recommendationKey': 'buy',
 'grossProfits': 115856000000,
 'freeCashflow': 46479876096,
 'targetMedianPrice': 370,
 'currentPrice': 280.72,
 'earningsGrowth': 0.222,
 'currentRatio': 2.247,
 'returnOnAssets': 0.15248999,
 'numberOfAnalystOpinions': 45,
 'targetMeanPrice': 368.33,
 'debtToEquity': 50.217,
 'returnOnEquity': 0.49051,
 'targetHighPrice': 411,
 'totalCash': 125348003840,
 'totalDebt': 80353001472,
 'totalRevenue': 184902991872,
 'totalCashPerShare': 16.72,
```

```
'financialCurrency': 'USD',
'revenuePerShare': 24.585,
'quickRatio': 2.05,
'recommendationMean': 1.7,
'exchange': 'NMS',
'shortName': 'Microsoft Corporation',
'longName': 'Microsoft Corporation',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'MSFT',
'messageBoardId': 'finmb_21835',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 10.867,
'beta3Year': None,
'enterpriseToEbitda': 22.122,
'52WeekChange': 0.07157314,
'morningStarRiskRating': None,
'forwardEps': 10.73,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 7496869888,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 21.335,
'sharesShort': 36814442,
'sharesPercentSharesOut': 0.0049,
'fundFamily': None,
'lastFiscalYearEnd': 1625011200,
'heldPercentInstitutions': 0.71901,
'netIncomeToCommon': 71184998400,
'trailingEps': 9.389,
'lastDividendValue': 0.62,
'SandP52WeekChange': 0.0261302,
'priceToBook': 13.157722,
'heldPercentInsiders': 0.00059,
'nextFiscalYearEnd': 1688083200,
'yield': None,
'mostRecentQuarter': 1640908800,
'shortRatio': 1.15,
'sharesShortPreviousMonthDate': 1646006400,
'floatShares': 7489894342,
'beta': 0.908333,
'enterpriseValue': 2009371246592,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 1045526400,
'lastSplitFactor': '2:1',
'legalType': None,
'lastDividendDate': 1644969600,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': 0.214,
'priceToSalesTrailing12Months': 11.381759,
'dateShortInterest': 1648684800,
'pegRatio': 1.88,
'ytdReturn': None,
'forwardPE': 26.162163,
'lastCapGain': None,
'shortPercentOfFloat': 0.0049,
'sharesShortPriorMonth': 41488998,
'impliedSharesOutstanding': 0,
```

```
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 274.03,
'regularMarketOpen': 273.29,
'twoHundredDayAverage': 305.08316,
'trailingAnnualDividendYield': 0.008612195,
'payoutRatio': 0.2449,
'volume24Hr': None,
'regularMarketDayHigh': 281.11,
'navPrice': None,
'averageDailyVolume10Day': 27597690,
'regularMarketPreviousClose': 274.03,
'fiftyDayAverage': 293.7042,
'trailingAnnualDividendRate': 2.36,
'open': 273.29,
'toCurrency': None,
'averageVolume10days': 27597690,
'expireDate': None,
'algorithm': None,
'dividendRate': 2.48,
'exDividendDate': 1652832000,
'circulatingSupply': None,
'startDate': None,
'regularMarketDayLow': 270.78,
'currency': 'USD',
'trailingPE': 29.898819,
'regularMarketVolume': 35522549,
'lastMarket': None,
'maxSupply': None,
'openInterest': None,
'marketCap': 2104521261056,
'volumeAllCurrencies': None,
'strikePrice': None,
'averageVolume': 33236583,
'dayLow': 270.78,
'ask': 0,
'askSize': 1000,
'volume': 35522549,
'fiftyTwoWeekHigh': 349.67,
'fromCurrency': None,
'fiveYearAvgDividendYield': 1.32,
'fiftyTwoWeekLow': 238.07,
'bid': 0,
'tradeable': False,
'dividendYield': 0.0091,
'bidSize': 1100,
'dayHigh': 281.11,
'regularMarketPrice': 280.72,
'preMarketPrice': None,
'logo_url': 'https://logo.clearbit.com/microsoft.com',
'trailingPegRatio': 1.8747}
```


In [3]:

```
# get historical market data
msft.history(period="max")
```

Out[3]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
1986-03-13	0.055783	0.063987	0.055783	0.061252	1031788800	0.0	0.0
1986-03-14	0.061252	0.064533	0.061252	0.063439	308160000	0.0	0.0
1986-03-17	0.063439	0.065080	0.063439	0.064533	133171200	0.0	0.0
1986-03-18	0.064533	0.065080	0.062345	0.062892	67766400	0.0	0.0
1986-03-19	0.062892	0.063439	0.061252	0.061799	47894400	0.0	0.0
...	...	...	...	...	...	...	...
2022-04-19	279.380005	286.170013	278.410004	285.299988	22297700	0.0	0.0
2022-04-20	289.399994	289.700012	285.369995	286.359985	22906700	0.0	0.0
2022-04-21	288.579987	293.299988	280.059998	280.809998	29454600	0.0	0.0
2022-04-22	281.679993	283.200012	273.380005	274.029999	29379300	0.0	0.0
2022-04-25	273.290009	281.109985	270.769989	280.720001	35609500	0.0	0.0

9104 rows × 7 columns

In [4]:
```
# show actions (dividends, splits)
msft.actions
```

Out[4]:

Date	Dividends	Stock Splits
1987-09-21	0.00	2.0
1990-04-16	0.00	2.0
1991-06-27	0.00	1.5
1992-06-15	0.00	1.5
1994-05-23	0.00	2.0
...	...	...
2021-02-17	0.56	0.0
2021-05-19	0.56	0.0
2021-08-18	0.56	0.0
2021-11-17	0.62	0.0
2022-02-16	0.62	0.0

82 rows × 2 columns

In [5]:
```
# show major holders
msft.major_holders
```

Out[5]:

	0	1
0	0.06%	% of Shares Held by All Insider
1	71.90%	% of Shares Held by Institutions
2	71.94%	% of Float Held by Institutions
3	5861	Number of Institutions Holding Shares

In [6]:
```
# show institutional holders
msft.institutional_holders
```

Out[6]:

	Holder	Shares	Date Reported	% Out	Value
0	Vanguard Group, Inc. (The)	615950062	2021-12-30	0.0822	207156324851
1	Blackrock Inc.	519035634	2021-12-30	0.0692	174562064426
2	State Street Corporation	302541869	2021-12-30	0.0404	101750881382
3	FMR, LLC	215377233	2021-12-30	0.0287	72435671002
4	Price (T.Rowe) Associates Inc	204196901	2021-12-30	0.0272	68675501744
5	Geode Capital Management, LLC	129107118	2021-12-30	0.0172	43421305925
6	Capital World Investors	107717797	2021-12-30	0.0144	36227649487
7	Capital Research Global Investors	92868182	2021-12-30	0.0124	31233426970
8	Capital International Investors	90985531	2021-12-30	0.0121	30600253785
9	Northern Trust Corporation	88410999	2021-12-30	0.0118	29734387183

In [7]:
```
# show analysts recommendations
msft.recommendations
```

Out[7]:

Date	Firm	To Grade	From Grade	Action
2012-03-16 08:19:00	Argus Research	Buy		up
2012-03-19 14:00:00	Hilliard Lyons	Long-Term Buy		main
2012-03-22 07:03:00	Morgan Stanley	Overweight		main
2012-04-03 11:53:00	UBS	Buy		main
2012-04-20 06:18:00	Deutsche Bank	Buy		main
...	...	...	...	...
2022-01-26 15:03:00	Morgan Stanley	Overweight		main
2022-01-26 15:06:01	Citigroup	Buy		main
2022-02-04 16:31:41	Tigress Financial	Buy		main
2022-04-19 11:57:30	Wells Fargo	Overweight		main
2022-04-19 13:28:16	Citigroup	Buy		main

368 rows × 4 columns

In [8]:
```
# show financials
msft.financials
msft.quarterly_financials

# show balance sheet
msft.balance_sheet
msft.quarterly_balance_sheet

# show cashflow
msft.cashflow
msft.quarterly_cashflow

# show earnings
msft.earnings
msft.quarterly_earnings
```

Out[8]:

Quarter	Revenue	Earnings
1Q2021	41706000000	15457000000
2Q2021	46152000000	16458000000
3Q2021	45317000000	20505000000
4Q2021	51728000000	18765000000

In [9]:
```
# show sustainability
msft.sustainability
```

Out[9]:

2022-2	Value
palmOil	False
controversialWeapons	False
gambling	False
socialScore	8.06
nuclear	False
furLeather	False
alcoholic	False
gmo	False
catholic	False
socialPercentile	None
peerCount	105
governanceScore	4.73
environmentPercentile	None
animalTesting	False
tobacco	False
totalEsg	13.26
highestControversy	3
esgPerformance	UNDER_PERF
coal	False
pesticides	False
adult	False
percentile	6.32
peerGroup	Software & Services
smallArms	False
environmentScore	0.46
governancePercentile	None
militaryContract	False

Available paramaters for the history( ) method are:

- **period**: data period to download (Either Use period parameter or use start and end)
  - Valid periods are: 1d, 5d, 1mo, 3mo, 6mo, 1y, 2y, 5y, 10y, ytd, max
- **interval**: data interval (intraday data cannot extend last 60 days)
  - Valid intervals are: 1m, 2m, 5m, 15m, 30m, 60m, 90m, 1h, 1d, 5d, 1wk, 1mo, 3mo
- **start**: If not using period - Download start date string (YYYY-MM-DD) or datetime.
- **end**: If not using period - Download end date string (YYYY-MM-DD) or datetime.
- **prepost**: Include Pre and Post market data in results? (Default is False)
- **auto_adjust**: Adjust all OHLC automatically? (Default is True)
- **actions**: Download stock dividends and stock splits events? (Default is True)

Mass download of market data

yfinance returns a pandas.DataFrame with multi-level column names, with a level for the ticker and a level for the stock price data.

You can also download data for multiple tickers at once, like before.

In [10]:
```
data = yf.download("AAPL TSLA 600519.SS 0700.HK", start="2019-01-01", end="2

data
```

```
[*********************100%***********************]  4 of 4 completed
```

Out[10]:

	Adj Close						
	0700.HK	600519.SS	AAPL	TSLA	0700.HK	600519.SS	
Date							
2018-12-31	NaN	NaN	38.233902	66.559998	NaN	NaN	39.4
2019-01-02	304.080292	577.981567	38.277523	62.023998	306.600006	598.979980	39.4
2019-01-03	302.493408	569.316467	34.464802	60.071999	305.000000	590.000000	35.5
2019-01-04	308.047394	580.895752	35.936077	63.537998	310.600006	602.000000	37.0
2019-01-07	314.989899	584.263367	35.856091	66.991997	317.600006	605.489990	36.9
...	...	...	...	...	...	...	
2021-12-24	460.799988	2194.090088	NaN	NaN	460.799988	2194.090088	
2021-12-27	NaN	2131.820068	180.100540	1093.939941	NaN	2131.820068	180.3
2021-12-28	450.399994	2138.179932	179.061859	1088.469971	450.399994	2138.179932	179.2
2021-12-29	444.799988	2041.000000	179.151749	1086.189941	444.799988	2041.000000	179.3
2021-12-30	443.399994	2075.000000	177.973251	1070.339966	443.399994	2075.000000	178.1

780 rows × 24 columns

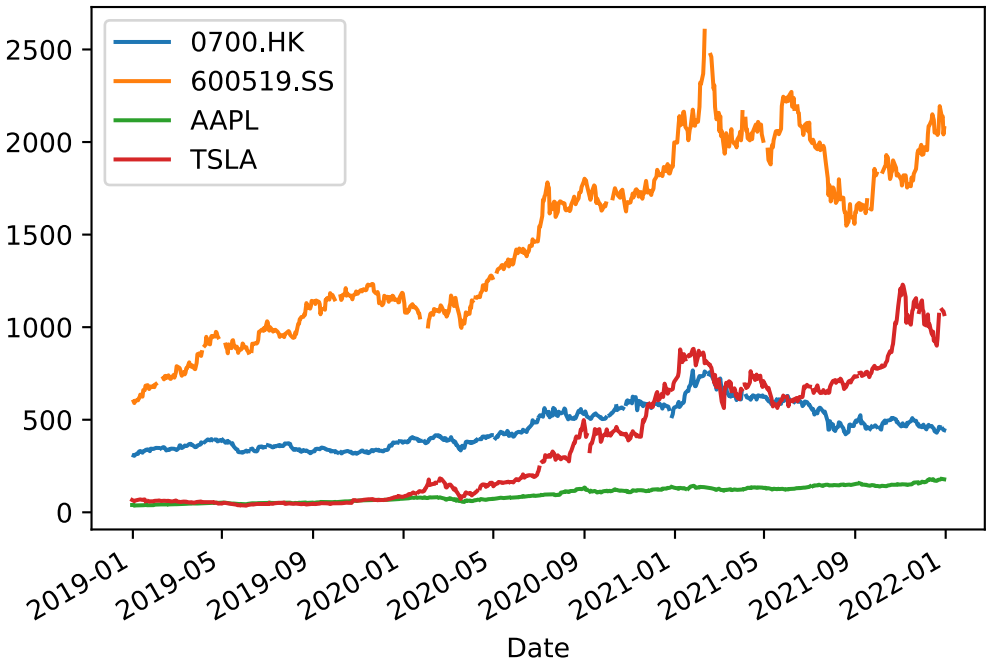
To access the closing price data for **MSFT**, you should use:

In [11]:
```
data['Close']['600519.SS']
```

Out[11]:
```
Date
2018-12-31            NaN
2019-01-02     598.979980
2019-01-03     590.000000
2019-01-04     602.000000
2019-01-07     605.489990
                  ...
2021-12-24    2194.090088
2021-12-27    2131.820068
2021-12-28    2138.179932
2021-12-29    2041.000000
2021-12-30    2075.000000
Name: 600519.SS, Length: 780, dtype: float64
```

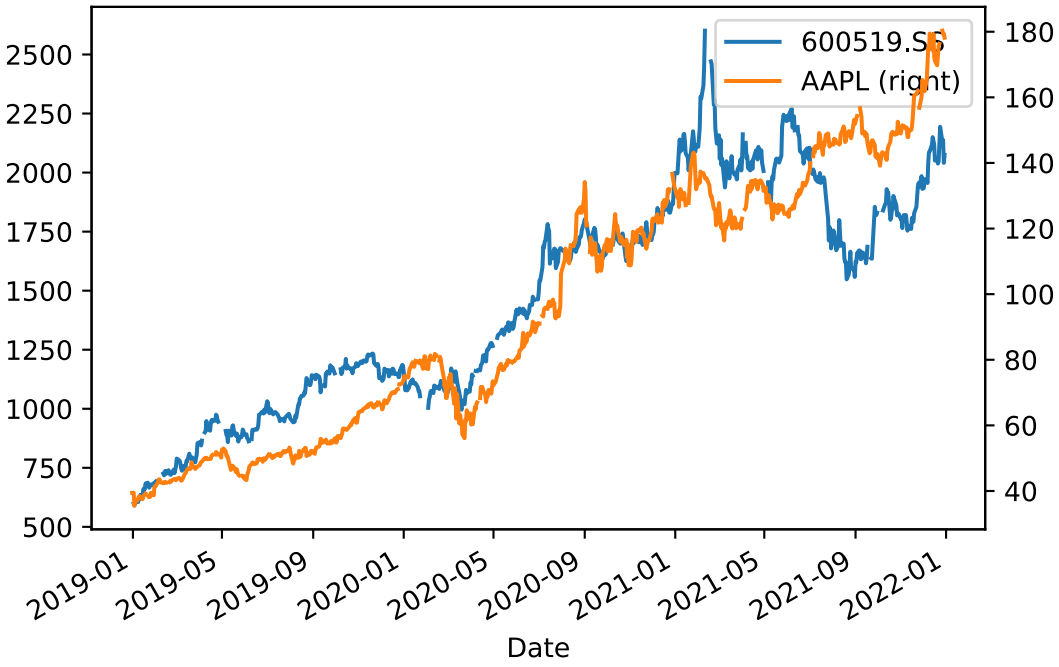
In [12]:
```
%matplotlib inline
%config InlineBackend.figure_format = 'svg'

data['Close'].plot()
```

Out[12]:
```
<AxesSubplot:xlabel='Date'>
```



In [13]:
```
data['Close'][['600519.SS','AAPL']].plot(secondary_y=['AAPL'])
```

Out[13]:
```
<AxesSubplot:xlabel='Date'>
```



World Bank

world_bank_data is an implementation of the **World Bank API** in Python.

Use this package to explore the World Development Indicators published by the World Bank.

Install **world_bank_data** using pip: $ **pip install world_bank_data**

Get the list of sources, topics, countries, regions

In [14]:
```
import world_bank_data as wb

#The list of topics is available with
wb.get_topics()
```

Out[14]:

id	value	sourceNote
1	Agriculture & Rural Development	For the 70 percent of the world's poor who liv...
2	Aid Effectiveness	Aid effectiveness is the impact that aid has i...
3	Economy & Growth	Economic growth is central to economic develop...
4	Education	Education is one of the most powerful instrume...
5	Energy & Mining	The world economy needs ever-increasing amount...
6	Environment	Natural and man-made environmental resources –...
7	Financial Sector	An economy's financial markets are critical to...
8	Health	Improving health is central to the Millennium ...
9	Infrastructure	Infrastructure helps determine the success of ...
10	Social Protection & Labor	The supply of labor available in an economy in...
11	Poverty	For countries with an active poverty monitorin...
12	Private Sector	Private markets drive economic growth, tapping...
13	Public Sector	Effective governments improve people's standar...
14	Science & Technology	Technological innovation, often fueled by gove...
15	Social Development	Data here cover child labor, gender issues, re...
16	Urban Development	Cities can be tremendously efficient. It is ea...
17	Gender	Gender equality is a core development objectiv...
18	Millenium development goals	
19	Climate Change	Climate change is expected to hit developing c...
20	External Debt	Debt statistics provide a detailed picture of ...
21	Trade	Trade is a key means to fight poverty and achi...

In [15]:
```
#Sources are returned by
wb.get_sources()
```

Out[15]:

id	lastupdated	name	code	description	url	dataavailability	metadataavailabili
1	2021-08-18	Doing Business	DBS			Y	
2	2022-04-08	World Development Indicators	WDI			Y	
3	2021-09-27	Worldwide Governance Indicators	WGI			Y	
5	2016-03-21	Subnational Malnutrition Database	SNM			Y	
6	2022-01-14	International Debt Statistics	IDS			Y	
...	...	...	...	...	...	...	
84	2021-05-11	Education Policy	EDP			Y	
85	2022-03-28	PEFA_2021_SNG	SNG			Y	
86	2021-09-24	Global Jobs Indicators Database (JOIN)	JON			Y	
87	2022-03-30	Country Climate and Development Report (CCDR)	CCD			Y	
88	2022-01-31	Food Prices for Nutrition	FPN			Y	

68 rows × 8 columns

In [16]:
```
#And finally, the list of countries is accessible with

wb.get_countries()
```

Out[16]:

id	iso2Code	name	region	adminregion	incomeLevel	lendingType	capitalCity
ABW	AW	Aruba	Latin America & Caribbean		High income	Not classified	Oranjestad
AFE	ZH	Africa Eastern and Southern	Aggregates		Aggregates	Aggregates	
AFG	AF	Afghanistan	South Asia	South Asia	Low income	IDA	Kabul
AFR	A9	Africa	Aggregates		Aggregates	Aggregates	
AFW	ZI	Africa Western and Central	Aggregates		Aggregates	Aggregates	
...	...	...	...	...	...	...	...
XZN	A5	Sub-Saharan Africa excluding South Africa and ...	Aggregates		Aggregates	Aggregates	
YEM	YE	Yemen, Rep.	Middle East & North Africa	Middle East & North Africa (excluding high inc...	Low income	IDA	Sana'a
ZAF	ZA	South Africa	Sub-Saharan Africa	Sub-Saharan Africa (excluding high income)	Upper middle income	IBRD	Pretoria
ZMB	ZM	Zambia	Sub-Saharan Africa	Sub-Saharan Africa (excluding high income)	Lower middle income	IDA	Lusaka
ZWE	ZW	Zimbabwe	Sub-Saharan Africa	Sub-Saharan Africa (excluding high income)	Lower middle income	Blend	Harare

299 rows × 9 columns

Get the list of indicators

This is done with the **get_indicators** function. You may query only the indicators for a specific source or topic as below.

If you input no arguments, the **get_indicator** function will return the description of all the 16,000+ indicators.

In [17]:
```
# topic and source id are from get_topics/get_sources
```

```
wb.get_indicators(topic=3, source=2)
```

Out[17]:

id	name	unit	source	sourceNote	sourceOrganizatio
BG.GSR.NFSV.GD.ZS	Trade in services (% of GDP)		World Development Indicators	Trade in services is the sum of service export...	Internationa Monetary Func Balance of Paymen.
BM.GSR.CMCP.ZS	Communications, computer, etc. (% of service i...		World Development Indicators	Communications, computer, information, and oth...	Internationa Monetary Func Balance of Paymen.
BM.GSR.FCTY.CD	Primary income payments (BoP, current US$)		World Development Indicators	Primary income payments refer to employee comp...	Internationa Monetary Func Balance of Paymen.
BM.GSR.GNFS.CD	Imports of goods and services (BoP, current US$)		World Development Indicators	Imports of goods and services comprise all tra...	Internationa Monetary Func Balance of Paymen.
BM.GSR.INSF.ZS	Insurance and financial services (% of service...		World Development Indicators	Insurance and financial services cover various...	Internationa Monetary Func Balance of Paymen.
...	...	...	...	...	.
NY.TTF.GNFS.KN	Terms of trade adjustment (constant LCU)		World Development Indicators	The terms of trade effect equals capacity to i...	World Bank nationa accounts data, an OECD Na.
PA.NUS.ATLS	DEC alternative conversion factor (LCU per US$)		World Development Indicators	The DEC alternative conversion factor is the u...	Internationa Monetary Func International Fin.
PA.NUS.PPP	PPP conversion factor, GDP (LCU per internatio...		World Development Indicators	Purchasing power parity (PPP) conversion facto...	Internationa Compariso Program, Worl Bank |.
PA.NUS.PPPC.RF	Price level ratio of PPP conversion factor (GD...		World Development Indicators	Price level ratio is the ratio of a purchasing...	Internationa Compariso Program, Worl Bank |.
PA.NUS.PRVT.PP	PPP conversion factor, private consumption (LC...		World Development Indicators	Purchasing power parity (PPP) conversion facto...	Internationa Compariso Program, Worl Bank |.

254 rows × 6 columns

Searching for one country or indicator

Use the functions **search_countries**, **search_source**, **search_indicators**.

Or, if you want to search in a existing dataframe, simply use search.

In [18]:
```
wb.search_indicators('GDP')
#wb.search_countries('China')
```

Out[18]:

id	name	unit	source	sourceNote	sourceOrganization
5.51.01.10.gdp	Per capita GDP growth		Statistical Capacity Indicators	GDP per capita is the sum of gross value added...	World Development Indicator (WDI) databank. Or...
6.0.GDP_current	GDP (current $)		LAC Equity Lab	GDP is the sum of gross value added by all res...	World Development Indicators (World Bank)
6.0.GDP_growth	GDP growth (annual %)		LAC Equity Lab	Annual percentage growth rate of GDP at market...	World Development Indicators (World Bank)
6.0.GDP_usd	GDP (constant 2005 $)		LAC Equity Lab	GDP is the sum of gross value added by all res...	World Development Indicators (World Bank)
6.0.GDPpc_constant	GDP per capita, PPP (constant 2011 internation...		LAC Equity Lab	GDP per capita based on purchasing power parit...	World Development Indicators (World Bank)
...	...	...	...	...	...
UIS.XUNIT.GDPCAP.23.FSGOV	Initial government funding per secondary stude...		Education Statistics	Total general (local, regional and central, cu...	UNESCO Institute for Statistics
UIS.XUNIT.GDPCAP.23.FSHH	Initial household funding per secondary studen...		Education Statistics	Total payments of households (pupils, students...	UNESCO Institute for Statistics
UIS.XUNIT.GDPCAP.3.FSGOV	Initial government funding per upper secondary...		Education Statistics	Total general (local, regional and central, cu...	UNESCO Institute for Statistics
UIS.XUNIT.GDPCAP.5T8.FSGOV	Initial government funding per tertiary studen...		Education Statistics	Total general (local, regional and central, cu...	UNESCO Institute for Statistics
UIS.XUNIT.GDPCAP.5T8.FSHH	Initial household funding per tertiary student...		Education Statistics	Total payments of households (pupils, students...	UNESCO Institute for Statistics

651 rows × 6 columns

Get the values of an indicator

The function **get_series** returns the value of a single indicator.

The World Bank API accepts quite a few arguments, including:

- mrv, integer: one or more most recent values
- date, string: either one year, or two years separated with a colon, like '2010:2018'
- gapfill, string: 'Y' or 'N' (the default): forward fills missing values.

For instance, the call below returns the most recent estimate for the World Population:

In [19]:
```
wb.get_series('SP.POP.TOTL', mrv=1)
```

Out[19]:
```
Country                          Series             Year
Africa Eastern and Southern      Population, total  2020    677243299.0
Africa Western and Central       Population, total  2020    458803476.0
Arab World                       Population, total  2020    436080728.0
Caribbean small states           Population, total  2020      7442291.0
Central Europe and the Baltics   Population, total  2020    102253057.0
                                                               ...
Virgin Islands (U.S.)            Population, total  2020       106290.0
West Bank and Gaza               Population, total  2020      4803269.0
Yemen, Rep.                      Population, total  2020     29825968.0
Zambia                           Population, total  2020     18383956.0
Zimbabwe                         Population, total  2020     14862927.0
Name: SP.POP.TOTL, Length: 266, dtype: float64
```

The result above has a 3-dimensional index.

Use the argument **simplify_index** to ignore the dimensions that take a single value (here: year and series). Also, use the argument **id_or_value='id'** if you prefer your data to be indexed by the codes rather than labels:

In [20]:
```
wb.get_series('SP.POP.TOTL', date='2016', id_or_value='id', simplify_index=T
```

Out[20]:
```
Country
AFE    609978946.0
AFW    412551299.0
ARB    404042892.0
CSS      7269385.0
CEB    102994278.0
           ...
VIR       107516.0
PSE      4367088.0
YEM     27168210.0
ZMB     16363449.0
ZWE     14030338.0
Name: SP.POP.TOTL, Length: 266, dtype: float64
```

Ready for an interative tutorial?

In [21]:
```
# Countries and associated regions
countries = wb.get_countries()
countries
```

Out[21]:

id	iso2Code	name	region	adminregion	incomeLevel	lendingType	capitalCity
ABW	AW	Aruba	Latin America & Caribbean		High income	Not classified	Oranjestad
AFE	ZH	Africa Eastern and Southern	Aggregates		Aggregates	Aggregates	
AFG	AF	Afghanistan	South Asia	South Asia	Low income	IDA	Kabu
AFR	A9	Africa	Aggregates		Aggregates	Aggregates	
AFW	ZI	Africa Western and Central	Aggregates		Aggregates	Aggregates	
...	...	...	...	...	...	...	..
XZN	A5	Sub-Saharan Africa excluding South Africa and ...	Aggregates		Aggregates	Aggregates	
YEM	YE	Yemen, Rep.	Middle East & North Africa	Middle East & North Africa (excluding high inc...	Low income	IDA	Sana'a
ZAF	ZA	South Africa	Sub-Saharan Africa	Sub-Saharan Africa (excluding high income)	Upper middle income	IBRD	Pretoria
ZMB	ZM	Zambia	Sub-Saharan Africa	Sub-Saharan Africa (excluding high income)	Lower middle income	IDA	Lusaka
ZWE	ZW	Zimbabwe	Sub-Saharan Africa	Sub-Saharan Africa (excluding high income)	Lower middle income	Blend	Harare

299 rows × 9 columns

In [22]:
```
# Population dataset, by the World Bank (most recent value)

population = wb.get_series('SP.POP.TOTL', id_or_value='id', simplify_index=T
population
```

Out[22]:
```
Country
AFE    677243299.0
AFW    458803476.0
ARB    436080728.0
CSS      7442291.0
CEB    102253057.0
           ...
VIR       106290.0
PSE      4803269.0
YEM     29825968.0
ZMB     18383956.0
ZWE     14862927.0
Name: SP.POP.TOTL, Length: 266, dtype: float64
```

In [23]:
```
# Aggregate region, country and population

df = countries[['region', 'name']].rename(columns={'name': 'country'}).loc[c
df['population'] = population

df
```

Out[23]:

id	region	country	population
ABW	Latin America & Caribbean	Aruba	106766.0
AFG	South Asia	Afghanistan	38928341.0
AGO	Sub-Saharan Africa	Angola	32866268.0
ALB	Europe & Central Asia	Albania	2837743.0
AND	Europe & Central Asia	Andorra	77265.0
...	...	...	...
XKX	Europe & Central Asia	Kosovo	1775378.0
YEM	Middle East & North Africa	Yemen, Rep.	29825968.0
ZAF	Sub-Saharan Africa	South Africa	59308690.0
ZMB	Sub-Saharan Africa	Zambia	18383956.0
ZWE	Sub-Saharan Africa	Zimbabwe	14862927.0

218 rows × 3 columns

In [24]:
```
import pandas as pd

# The sunburst plot requires weights (values), labels, and parent (region, c
# We build the corresponding table here

columns = ['parents', 'labels', 'values']

level1 = df.copy()

# rename colomns
level1.columns = columns
# number formatting
level1['text'] = level1['values'].apply(lambda pop: '{:,.0f}'.format(pop))

level2 = df.groupby('region')['population'].sum().reset_index()[['region', '
level2.columns = columns
level2['parents'] = 'World'
level2['text'] = level2['values'].apply(lambda pop: '{:,.0f}'.format(pop))
```

```
level2['values'] = 0

level3 = pd.DataFrame({'parents': [''], 'labels': ['World'],
                       'values': [0.0], 'text': ['{:,.0f}'.format(population

all_levels = pd.concat([level1, level2, level3], axis=0).reset_index(drop=Tr
all_levels
```

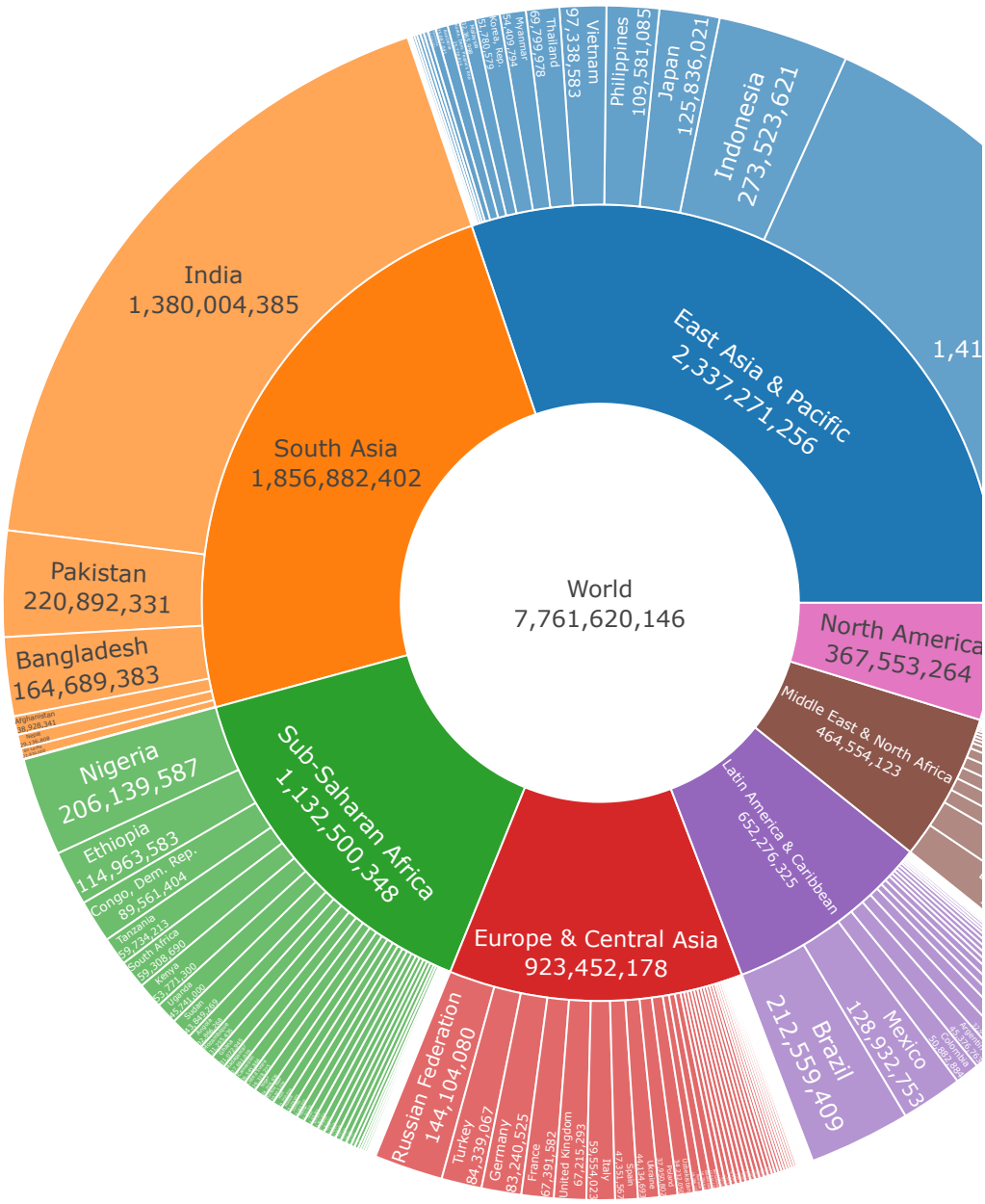
Out[24]:

	parents	labels	values	text
0	Latin America & Caribbean	Aruba	106766.0	106,766
1	South Asia	Afghanistan	38928341.0	38,928,341
2	Sub-Saharan Africa	Angola	32866268.0	32,866,268
3	Europe & Central Asia	Albania	2837743.0	2,837,743
4	Europe & Central Asia	Andorra	77265.0	77,265
...	...	...	...	...
221	World	Middle East & North Africa	0.0	464,554,123
222	World	North America	0.0	367,553,264
223	World	South Asia	0.0	1,856,882,402
224	World	Sub-Saharan Africa	0.0	1,132,500,348
225		World	0.0	7,761,620,146

226 rows × 4 columns

In [25]:

```
# And now we can plot the World Population

import plotly
import plotly.offline as offline

offline.iplot(dict(data=[dict(type='sunburst', hoverinfo='text', **all_level
                   layout=dict(title='World Population (World Bank, 2017)<br
                               width=800, height=800)),
              validate=False)
```


World Population (World Bank, 2017)
Click on a region to zoom
World
7,761,620,146
East Asia & Pacific
2,337,271,256
South Asia
1,856,882,402
Sub-Saharan Africa
1,132,500,348
Europe & Central Asia
923,452,178
Latin America & Caribbean
652,276,325
Middle East & North Africa
464,554,123
North America
367,553,264
Indonesia
273,523,621
Philippines
109,581,085
Vietnam
97,338,583
Thailand
69,799,978
India
1,380,004,385
Pakistan
220,892,331
Bangladesh
164,689,383
Nigeria
206,139,587
Ethiopia
114,963,583
Congo, Dem. Rep.
89,561,404
Russian Federation
144,104,080
Turkey
84,339,067
Germany
83,240,525
France
67,391,582
United Kingdom
67,215,293
Brazil
212,559,409
Mexico
128,932,753

In [ ]: