Under the hood with Altair

In this notebook, we will examine Altair under the hood (https://altair-viz.github.io/user_guide/internals.html). The Altair docs specify that Altair is a "Python API for generating validated Vega-Lite specifications". That's it!

- So you can think of Altair as a Python wrapper for Vega-Lite.
- Vega lite is a declarative specification for graphics. It's a grammar, like we discussed on Friday.

In [1]:
```
! pip install vega_datasets # to get started
```

```
Traceback (most recent call last):
  File "/Users/jiangxinyu/anaconda3/bin/pip", line 7, in <module>
    from pip._internal.cli.main import main
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/cli/main.py", line 8, in <module>
    from pip._internal.cli.autocompletion import autocomplete
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/cli/autocompletion.py", line 9, in <module>
    from pip._internal.cli.main_parser import create_main_parser
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/cli/main_parser.py", line 7, in <module>
    from pip._internal.cli import cmdoptions
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/cli/cmdoptions.py", line 22, in <module>
    from pip._internal.cli.progress_bars import BAR_TYPES
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/cli/progress_bars.py", line 9, in <module>
    from pip._internal.utils.logging import get_indentation
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/utils/logging.py", line 14, in <module>
    from pip._internal.utils.misc import ensure_dir
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/utils/misc.py", line 29, in <module>
    from pip._internal.locations import get_major_minor_version, site_packages, user_site
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/locations/__init__.py", line 9, in <module>
    from . import _distutils, _sysconfig
  File "/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/locations/_sysconfig.py", line 8, in <module>
    from pip._internal.exceptions import InvalidSchemeCombination, UserInstallationInvalid
ImportError: cannot import name 'InvalidSchemeCombination' from 'pip._internal.exceptions' (/Users/jiangxinyu/anaconda3/lib/python3.7/site-packages/pip/_internal/exceptions.py)
```

In [20]:
```
import altair as alt
from vega_datasets import data

chart = alt.Chart("https://cdn.jsdelivr.net/npm/vega-datasets@v1.29.0/data/cars.json").mark_bar().encode(
    x='Horsepower:Q',
    y='Miles_per_Gallon:Q',
    color='Origin:N',
).configure_view(
    continuousHeight=300,
    continuousWidth=500,
)

print(chart.to_json(indent=2))
```

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v4.17.0.json",
  "config": {
    "view": {
      "continuousHeight": 300,
      "continuousWidth": 500
    }
  },
  "data": {
    "url": "https://cdn.jsdelivr.net/npm/vega-datasets@v1.29.0/data/cars.json"
  },
  "encoding": {
    "color": {
      "field": "Origin",
      "type": "nominal"
    },
    "x": {
      "field": "Horsepower",
      "type": "quantitative"
    },
    "y": {
      "field": "Miles_per_Gallon",
      "type": "quantitative"
    }
  },
  "mark": "bar"
}
```

What are the fields in the dictionary above? What do you think they mean?

Generate the data with x axis equal to `Horsepower` and y axis equal to `Miles_per_Gallon` and set the color to `Origin` . Print is printing the jason to visiulize it.

Try changing the Altair code above. What do you notice is different in the output Vega specification?

Describe your findings here. Try a few different changes in new cells below here. Comment on your observations below. How do your changes in Altair cause changes in the Vega grammar? Try to be as specific as possible.

In [18]:
```
import pandas as pd  # one example to start

df = pd.DataFrame({'Horsepower': [1,2,3],
                   'Miles_per_Gallon': [-1, -2, -3]})

chart = alt.Chart(df).mark_bar().encode(
    x='Horsepower:O',
    y='Miles_per_Gallon:Q'
).configure_view(
    continuousHeight=300,
    continuousWidth=500,
)


print(chart.to_json(indent=2))
```


```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v4.17.0.json",
  "config": {
    "view": {
      "continuousHeight": 300,
      "continuousWidth": 500
    }
  },
  "data": {
    "name": "data-40af22ac401de99d2b3ca87883b39eed"
  },
  "datasets": {
    "data-40af22ac401de99d2b3ca87883b39eed": [
      {
        "Horsepower": 1,
        "Miles_per_Gallon": -1
      },
      {
        "Horsepower": 2,
        "Miles_per_Gallon": -2
      },
      {
        "Horsepower": 3,
        "Miles_per_Gallon": -3
      }
    ]
  },
  "encoding": {
    "x": {
      "field": "Horsepower",
      "type": "ordinal"
    },
    "y": {
      "field": "Miles_per_Gallon",
      "type": "quantitative"
    }
  },
  "mark": "bar"
}
```


Change from `mark_point` to `mark_bar`

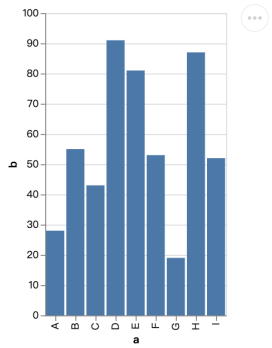
Altair to vega

Try copying your Altair json printout into the Vega lite (https://vega.github.io/editor/#/custom/vega-lite) editor. It should just work.

Alternately, you can render Vega json specifications as Altair charts

In [22]:
```
alt.Chart.from_json("""
{
  "$schema": "https://vega.github.io/schema/vega-lite/v4.json",
  "description": "A simple bar chart with embedded data.",
  "data": {
    "values": [
      {"a": "A","b": 28}, {"a": "B","b": 55}, {"a": "C","b": 43},
      {"a": "D","b": 91}, {"a": "E","b": 81}, {"a": "F","b": 53},
      {"a": "G","b": 19}, {"a": "H","b": 87}, {"a": "I","b": 52}
    ]
  },
  "mark": "bar",
  "encoding": {
    "x": {"field": "a", "type": "ordinal"},
    "y": {"field": "b", "type": "quantitative"}
  }
}
""")
```

Out[22]:



Altair gallery

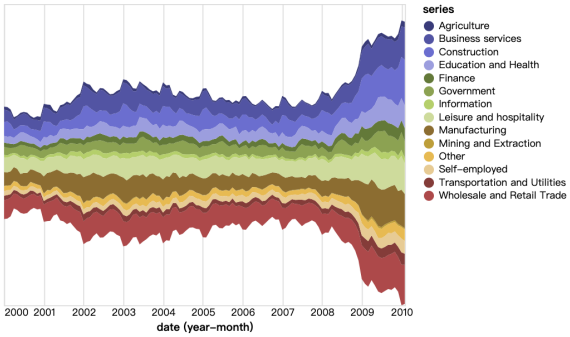
Try picking one or more complex examples from the Altair gallery (https://altair-viz.github.io/gallery/streamgraph.html). What do you think the corresponding vega specification will look like? Write out your expectations, and intrepretation of the Vega output. Is it more verbose or less verbose that you expected?

In [5]:
```
import altair as alt
from vega_datasets import data

source = data.unemployment_across_industries.url

alt.Chart(source).mark_area().encode(
    alt.X('yearmonth(date):T',
        axis=alt.Axis(format='%Y', domain=False, tickSize=0)
    ),
    alt.Y('sum(count):Q', stack='center', axis=None),
    alt.Color('series:N',
        scale=alt.Scale(scheme='category20b')
    )
).interactive()
```

Out[5]:



In [6]:
```
print(alt.Chart(source).mark_area().encode(
    alt.X('yearmonth(date):T',
        axis=alt.Axis(format='%Y', domain=False, tickSize=0)
    ),
    alt.Y('sum(count):Q', stack='center', axis=None),
    alt.Color('series:N',
        scale=alt.Scale(scheme='category20b')
    )
).to_json())


```


```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v4.17.0.json",
  "config": {
    "view": {
      "continuousHeight": 300,
      "continuousWidth": 400
    }
  },
  "data": {
    "url": "https://cdn.jsdelivr.net/npm/vega-datasets@v1.29.0/data/unemployment-across-industries.json"
  },
  "encoding": {
    "color": {
      "field": "series",
      "scale": {
        "scheme": "category20b"
      },
      "type": "nominal"
    },
    "x": {
      "axis": {
        "domain": false,
        "format": "%Y",
        "tickSize": 0
      },
      "field": "date",
      "timeUnit": "yearmonth",
      "type": "temporal"
    },
    "y": {
      "aggregate": "sum",
      "axis": null,
      "field": "count",
      "stack": "center",
      "type": "quantitative"
    }
  },
  "mark": "area"
}
```


In [ ]:
```

```

In [ ]:
```

```