Data sources

Allow integrations to provide data sources to be consumed by other sources. Integrations decide their own configuration. Data will be forwarded to other sources via variables. That way it will be easy to consume, share and avoid fetching duplicate data to source multiple sensors.

Data sources consumers can consume multiple sources.

~~Data source updates will be controlled by the data source. Data source can implement polling internally.~~ -> this is scrapped and should be done via triggers. We still want data sources to be loaded on demand.

Data source examples

- Fetch data from a URL
- Historical data from database
- Query InfluxDB
- Run forecast model on current data
- Run image recognition on camera
- Command line command (not a good idea, see danger heading)
- Query calendar events
- Detailed driving or route directions
- Query large language model

Using data sources

Data sources will be a new action in the script syntax. This makes it available to automation and scripts.

Data sources are defined like variables, except the value is executed by the referenced integration.

Data sources should be able to access variables available in the scope. Could render some part of their config with that.

```
script:
  notify_daily_energy:
  - data_sources:
      # Output of each data source is stored in variable
      # with same name as key in config
      today_usage:
        platform: recorder
        type: sum
        statistic: sensor.total_energy_usage
        period: today  # this_week, last_week, this_month
      forecast_usage:
        platform: energy_forecaster
        type: forecast
        statistic: sensor.total_energy_usage
        period: today
  - service: notify.paulus
    message:
      > Your usage today is {{ today_usage }} and we expect a total
of {{ forecast_usage }}.
```

Danger of data sources

If we allow doing command line stuff, it will allow UI access/blueprints to run arbitrary code. Probably not a good idea.

For our first version we should limit to things that only expose read only things.

Alternative idea by Frenck

The first time this idea came up, we've pivoted and implemented below instead:

The alternative idea is for triggers to be used to feed into template entities.

Standardize how templated entities are defined for integrations that need it. So rest can define multiple sensors etc.

```
template:
  - trigger:
     - platform: time_pattern
     # This will update every night
     hours: 0
     minutes: 0
     sensor:
     # Keep track how many days have past since a date
     - name: "Not smoking"
     state: '{{ ( ( as_timestamp(now()) -
as_timestamp(strptime("06.07.2018", "%d.%m.%Y")) ) / 86400 ) |
round(default=0) }}'
     unit_of_measurement: "Days"
```

Alternative: Service Calls

Discussed in this Architecture discussion, service calls are be too broad of an API, and also may encourage side effects (similar concern as wanting datasources to be read-only above)

Alternative: Template functions

Discussed in this Architecture discussion, could allow integration specific ways to extend templates directly with arbitrary routines. Doing I/O in a template also seems risky and perhaps difficult, has a similarly broadly open API, and may cause difficult to diagnose issues given templates are used everywhere.

Data source consumer example: template (old)

Legacy example, this is not the right syntax.

Template integration will add support for top level yaml to define source + templates that will consume the data of the source.

Extra: add variables block that allow user to define variables with source variables to make defining template entities easier.

```
template:
  sources:
    - platform: mqtt
      topic: home/status
      type: json
      variable: home_status
  sensor:
    version:
      friendly_name: Home Version
      value_template: {{ home_status.version }}
  binary_sensor:
    online:
      friendly_name: Home Online
      value_template: {{ home_status.online }}
```