UNIVERSITY
4 INDUSTRY

Einführung Plotly Dash

Robert Heise und Florian Edenhofner – Education4Industry GmbH

23.06.2024

University4Industry

Agenda

Kurzeinführung App-Entwicklung mit Ploty Dash

1. Was ist Plotly Dash?
2. Grundaufbau des Quellcodes
3. Beschreibung des Layouts als Komposition verschiedener Komponenten
4. Beschreibung der Funktionalität (Callbacks)
5. Erweiterte Beschreibung des Layouts mit Bootstrap-Komponenten

Was ist Plotly Dash?

Plotly Dash ist ein Framework speziell für die Entwicklung analytischer Dashboards als Webapplikationen.

- flexibles Framework für die Erstellung von Dashboards.
- Point&Click-Interface
- einfaches High-Level-Interface für Python
- natives Zusammenspiel mit Plotly Express
- Kenntnisse in Html, CSS und JavaScript sind nicht notwendig
- basierend auf Flask, Plotly.js, React.js
- Verfügbar für Python, R, Julia, F#

`https://dash.plotly.com`
Tutorials mit zahlreichen Beispielen

Grundaufbau des Quellcodes

Dash stellt als Framework Klassen und Funktionen zur Verfügung, welche es ermöglichen eine Webapplikation ohne hohen Programmieraufwand zu gestalten.

Typische Bestandteile einer Dash-Applikation:

```
1  from dash import Dash, html
2
3  # Inizialisierung der Dash-App
4  app = Dash()
5
6  # Beschreibung des Layouts - Wie soll die App im Browser aussehen?
7  app.layout = html.Div(...)
8
9  # Beschreibung einer Funktionalität (callbacks) mittels Dekoratoren
10 @app.callback(Output(...),Input(...))
11 def graph_update(...):
12   ...
13   return output_values
14
15 # Starten der Dash-App
16 if __name__ == '__main__':
17     app.run_server(debug=True) # Startet Server im Debug-Modus
```

Layout: Komposition aus Komponenten

Ein Layout wird aus verschiedenen Komponenten zusammengesetzt.

Eine Komponente als Elemente der Webseite zu stehen. Sie sind hierarchisch "verschachtelt".

- Überschriften, Textabschnitte
- Menüs
- Grafiken
- ...

Es gibt verschiedene Typen von Komponenten.

- Html-Komponenten
- Dash-Core-Komponenten
- Bootstrap-Kopmponenten
- ...

Layout: Html-Komponenten

HTML-Komponenten strukturieren eine Webseite.

- Sie beschrieben typische Komponenten, wie Überschriften, Buttons etc....
- Hierarchischer Aufbau
- Das Layout einer Dash-App kann entsprechend Html-Komponenten enthalten.

```
1  from dash import Dash, html
```

```
1  # Beipiel für das Erstellen eines Layouts der App ...
2  # ... durch das Zusammenfügen verschiedener HTML-Komponenten
3  app.layout = html.Div(
4      children=[
5          html.H1(children='Eine Beispiel-App',
6                  style={'textAlign': 'center', 'marginTop': 40, 'marginBottom': 40}),
7          html.H2(children='Wissenswertes'),
8          html.Div(children='Beipieltext. Dash ist ein tolles Ding!'),
9      ]
10 )
```

siehe auch `dash_example_1_html_components.py`

Dash Layout: weitere Html-Komponenten

Einige Beispiele für Html-Komponenten

```
1    html.Div()          # Division - Generische Html-Container
2    html.Button()       # Druckschaltfläche
3    html.H1()           # Überschrift Größen 1
4    html.H2()           # Überschrift Größen 2
5    html.H3()           # Überschrift Größen 3
6    html.Img()          # Bild
```

Weiter Information über HTML-Komponenten unter
https://dash.plotly.com/dash-html-components

Informationen zu einzelnen Html-Komponenten unter

https://developer.mozilla.org/en-US/docs/Web/HTML

Dash Layout: Dash core components

Dash core components erweitern die Html-Komponenten durch spezielle dash-eigene Objekte. Diese werden im Browser oft unter Verwendung von JavaScript dargestellt.

```
1  from dash import dcc # Import der Dash Core Components (dcc)
```

Layout mit Diagramm mittels dcc.Graph und plotly.express

```
1  import plotly.express as px
2  df = px.data.stocks()                                  # Beispieldaten als DataFrame
3  fig = px.line(df, x=df['date'], y=df['GOOG'])          # Liniendiagramm
4
5  app.layout = html.Div(
6      children=[
7          html.H1(children='Zeitreihe Aktie Google'), # Überschrift
8          dcc.Graph(figure=fig)                       # Graph-Komponente bettet Diagramm ein
9      ]
10 )
```

siehe auch `dash_example_2_dcc_plotly.py`

Dash Layout: weitere Core components

Einige Beispiele für weitere Komponenten

```
1  dcc.Graph()              # Container für Grafik
2  dcc.Input()              # Texteingabe
3  dcc.Dropdown()           # Dropdown-Menü erlaubt Auswahl von Optionen
4  dcc.Slider()             # Einstellen eines Wertes
5  dcc.Textarea()           # Darstellung von Text
6  dcc.Checklist()          # Liste von Checkboxen
7  dcc.RadioItems()         # Radioitems zur Auswahl von Optionen
8  dcc.DatePickerSingle()   # Auswahl eines Datums aus einer Kalenderansicht
```

Weiter Information über HTML-Komponenten unter

https://dash.plotly.com/dash-core-components

Dash Callbacks

Callbacks erlauben es u.a. Reaktionen der App auf mögliche User-Interaktionen zu defineren. Sie beschreiben:

- welche Aktion des Anwenders an welcher Komponente der Website
- welche Reaktion an welchem Komponente der Webseite zur Folge hat
- und welcher Programmcode dazu verwendet wird.

Einzelne Elemente des Layouts müssen dazu mittels eines Argumentes id identifiziert werden.

```
1  # Layoutbeschreibung mit Id's und Dropout-Element
2  app.layout = html.Div(
3      children=[
4          html.H3(id='title', children='Aktienkurse'),        # enthält id = 'title'
5          dcc.Dropdown(id='dropdown',                         # enthält id = 'dropdown'
6                       options=[
7                           {'label': 'Google', 'value': 'GOOG'},
8                           {'label': 'Apple', 'value': 'AAPL'},
9                           {'label': 'Amazon', 'value': 'AMZN'},
10                      ],
11                      value='GOOG'),
12         dcc.Graph(id='line_plot'),                          # enthält id = 'line_plot'
13     ]
14 )
```

siehe auch `dash_example_3_dcc_with_callbacks.py`

Dash Callbacks

Für die eigentliche Beschreibung des Callbacks werden Dekoratoren verwendet. Sie werden nach dem Layout erklärt und verknüpfen Input- und Outputkomponenten mittels ihrer Id.

```
1  from dash.dependencies import Input, Output
```

Beispiel für Callback

```
1  @app.callback(Output(component_id='line_plot', component_property= 'figure'),
2                Input(component_id='dropdown', component_property= 'value'))
3  def graph_update(value_of_input_component):
4      print(value_of_input_component)
5      fig = px.line(df, x = df['date'], y = df[value_of_input_component])
6      return fig
```

- Der Dekorator ist eine Function Factory.
- Output- und Inputkomponenten und deren Property werden definiert.
- Übergabe der Funktionsargument erfolgt in der Reihenfolge der Definition der Inputs.
- Der Rückgabewert der Funktion entspricht der neuen Property der Outputkomponente.

Dash Callbacks - vollständiger Code der App

```
1   from dash import Dash,dcc,html
2   from dash.dependencies import Input, Output
3   import plotly.express as px
4   df = px.data.stocks()
5   app = dash.Dash()
6
7   #Layoutbeschreibung mit Id's und Dropout-Element
8   app.layout = html.Div(children=[html.H3(id='H2', children='Aktienkurse'),
9                                   dcc.Dropdown(id='dropdown',
10                                      options=[{'label': 'Google', 'value': 'GOOG'},
11                                               {'label': 'Apple', 'value': 'AAPL'},
12                                              {'label': 'Amazon', 'value': 'AMZN'},],
13                                      value='GOOG'),
14                                  dcc.Graph(id='line_plot')])
15
16  # Callback erklärt Funktion des Dropdown-Element
17  @app.callback(Output(component_id='line_plot', component_property='figure'),
18                Input(component_id='dropdown', component_property='value'))
19  def graph_update(value_of_input_component):
20      print(value_of_input_component)
21      fig = px.line(df, x=df['date'], y=df[value_of_input_component])
22      return fig
23
24  if __name__ == '__main__':
25      app.run_server(debug=True)
```

dash_example_3_dcc_with_callbacks.py

Bootstrap-Komponeten

Dash-Bootstrap-Komponenten erweitern Dash-Apps um Bootstrap-CSS. Dadurch können u.a. z.B. Zeilen und Spalten erzeugt werden.

```
import dash_bootstrap_components as dbc
```

externen Stylesheets ändert Erscheinung der App

```
app = dash.Dash(external_stylesheets=[dbc.themes.BOOTSTRAP])
```

Layout mit Bootstrap-Komponenten Row und Col

```
app.layout = html.Div(
  children=[
    dbc.Row([                                  # Erstellung einer Zeile
      dbc.Col([html.H1('HTML Component')]),    # Erstellung einer Spalte in dieser Zeile
      dbc.Col([html.H1('HTML Component')]),
      ], align='center'),
  ]
)
```

siehe auch `dash_example_4_dbc_with_callbacks.py`

https://dash-bootstrap-components.opensource.faculty.ai/

Zusammenfassung Komponenten

Verschiedene Komponententypen

- Html components (HTML-Elemente)
- Dash core components (dash-eigene Elemente)
- Bootstrap components (Bootstrap-Element)
- Dash DAQ (weitere dash-eigegne Bedienelemente)

Es existieren noch mehr Pakete mit speziellen Komponenten.

`https://dash.plotly.com/`

Ändern von IP und Port

```
1    if __name__ == '__main__':
2        app.run_server(host = '0.0.0.0', port=8080, debug=False)
```

Eine auf dem Raspberry Pi laufende Dash-Applikation kann durch Angabe der IP-Adresse des Raspberry's im lokalen WLan freigegeben werden!

Das Auto kann auf diese Weise z.B. durch einen Klick auf einen Button gestartet oder gestoppt werden.

Tipps zum Einarbeiten in Dash

- Verstehen Sie den Grundaufbau
  - Layout + Callbacks (`https://dash.plotly.com/`)
- Verschaffen Sie sich einen Überblick über die verschiedenen Elemente
  - html components (`https://dash.plotly.com/dash-html-components`)
  - dash core components (`https://dash.plotly.com/dash-core-components`)
  - bootstrap components (`https://dash-bootstrap-components.opensource.faculty.ai/`)
- Einzelne Element haben eine Vielzahl von Stylingoptionen
  - Konzentrieren Sie sich zuerst nicht auf das Styling!
  - Erkunden Sie die Stylingoptionen nur für wenige ausgewählte Elemente