CSS transitions & animations

Examples page

In your project:

- Create a page **`examples.html`**
- Make sure the page has the correct html5 structure
- Start a new section with a heading for each exercise

Pseudo classes

Pseudo-classes

Define a state of an element

- `:hover` mouse over
- `:active` active links
- `:visited` visited links
- `:focus` focused element

:hover

Gets applied when mouse is over an element

```
.button {
    background-color: red;
}

.button:hover {
    background-color: yellow;
}
```

:hover

In the example page:

- Create a div that changes its background color to **`#ff00ff`** whenever the mouse is over

:hover

In your example page:

- Create a container div **`#father`**
  - width and height 500px
  - background color black
  - position relative
- Inside the container create div: **`#son`**
  - position absolute
  - width and height 200px
  - left 200px, top 200px
  - background color red
- When hovering **`#father`**: **`#son`** must change background color from red to green

:active

Gets applied to a link when you click on it

```
a {
    color: red;
}

a:active {
    color: yellow;
}
```

:active

In your **project**:

- Make **all the links** change their font color to **`#ffff00`** whenever you click
- Can you also remove the underline from the links when you click on it?

:visited

Gets applied to a link when it was already visited

```
a {
    color: red;
}

a:visited {
    color: red;
}
```

:visited

In your **project**:

- Make the visited links look exactly like the unvisited links

:focus

Gets applied to the input element that has focus

```
#username {
    border: none;
}

#username:focus {
    border: 1px solid red;
}
```

:focus

In your **project**:

- Make all the input fields change their background color whenever they are focused

Transformations

rotate

Rotates an element from its current position

```
#box1 {
    transform: rotate(45deg);
}

#box2 {
    transform: rotate(-45deg);
}
```

rotate

In the examples page:

- Create two divs: **`#div1`**, **`#div2`**
- Give them the same position and dimensions
- Give them two different semi-transparent background color
- Rotate **`#div1`** 20deg clockwise
- Rotate **`#div1`** 20deg counter-clockwise

scale

Increases or decreases the size of an element (according to the parameters given for the width and height).

```
#box1 {
    transform: scale(2, 2);
}

#box2 {
    transform: scale(0.5, 1);
}
```

scale

In the examples page:

- Create two divs: **`#div3`**, **`#div4`**
- Write some text in them
- Make **`#div3`** look three times as big
- Make **`#div4`** have half of the height and twice of the length

translate

Moves an element from its current position

```
#box1 {
    transform: translate(50px,50px);
}
```

Very important!

Transform properties should be "vendor-prefixed"

```
div {
  -webkit-transform: scale(1.5);
  -moz-transform: scale(1.5);
  -o-transform: scale(1.5);
  transform: scale(1.5);
}
```

Transitions

transition-duration

Indicates how long the transition will last

Default: zero seconds

```
.button {
    background-color: red;
    transition-duration: 1s;
}

.button:hover {
    background-color: green;
}
```

transition-duration

In your **project**:

- Make all links change font color on hover with a transition of 1 second duration

transition-property

Indicates which property will transition

Default: all

```
.button {
    background-color: red;
    transition-property: transform;
}

.button:hover {
    background-color: green;
    transform: scale(2);
}
```

transition-property

In the examples page:

- Create a div: **`#div5`**
- On hover,
  - rotate the div of 20 deg counterclockwise **with a transition**
  - change the background color **without a transition**

transition-property

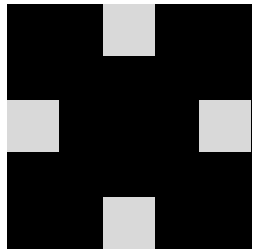
In the examples page:

- Create a div: **`#div6`**
- On hover,
  - rotate the div of 20 deg counterclockwise **without a transition**
  - change the background color **with a transition**

transition

In your examples page:

- Create the div **`#compass`**. It should look exactly like **`#father`**
- Inside **`#compass`**, create 4 divs: **`#north`**, **`#south`**, **`#west`**, **`#east`**
  - 100px width and length
  - white background color
- Position each one of them at the center of each side
- When hovering **`#compass`**, the 4 divs should translate (with transition!) to the center of **`#compass`**



transition-delay

How long a transition should be stalled before executing

Default: zero seconds

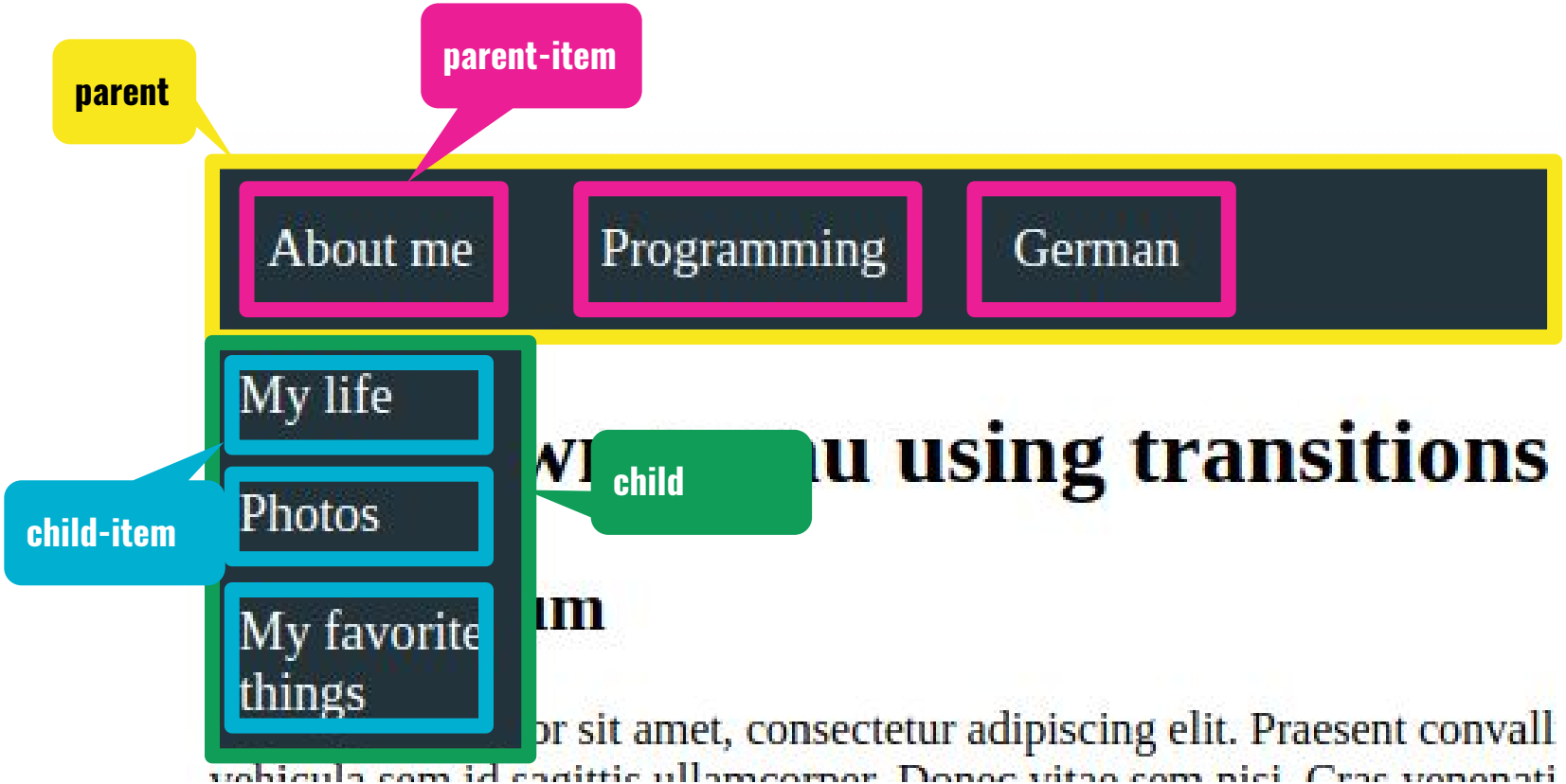
```
.button {
    background-color: red;
    transition-delay: 1s;
}

.button:hover {
    background-color: green;
}
```

transition-delay

In the examples page:

- Change **`#north`**, **`#south`**, **`#west`**, **`#east`** transitions so that:
  - **`#north`** starts immediately
  - **`#east`** starts after 0.5 seconds
  - **`#south`** starts after 1 second
  - **`#west`** starts after 1.5 seconds

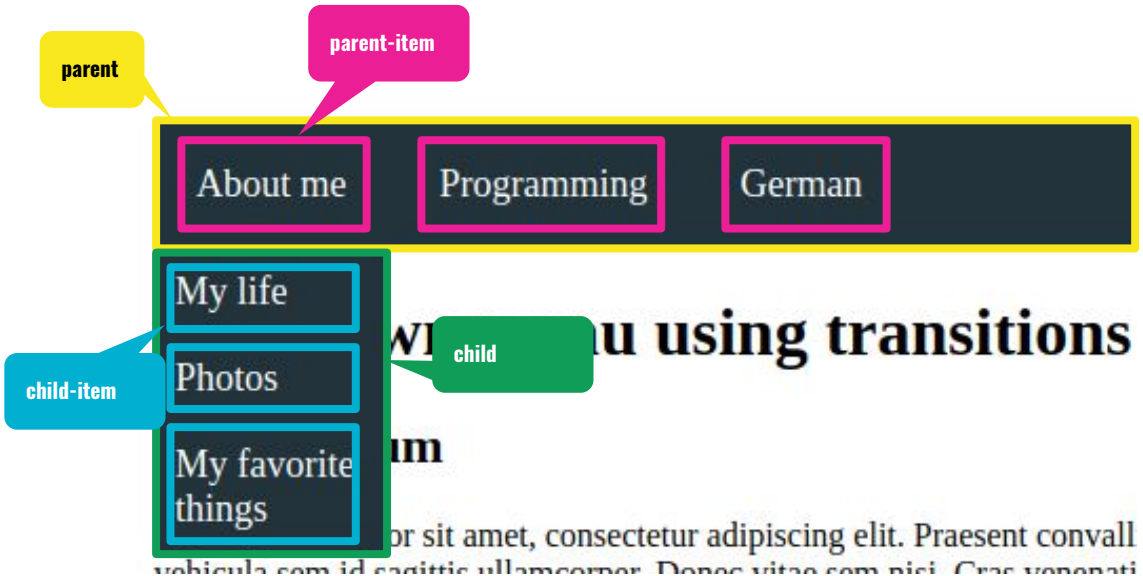
Project time!

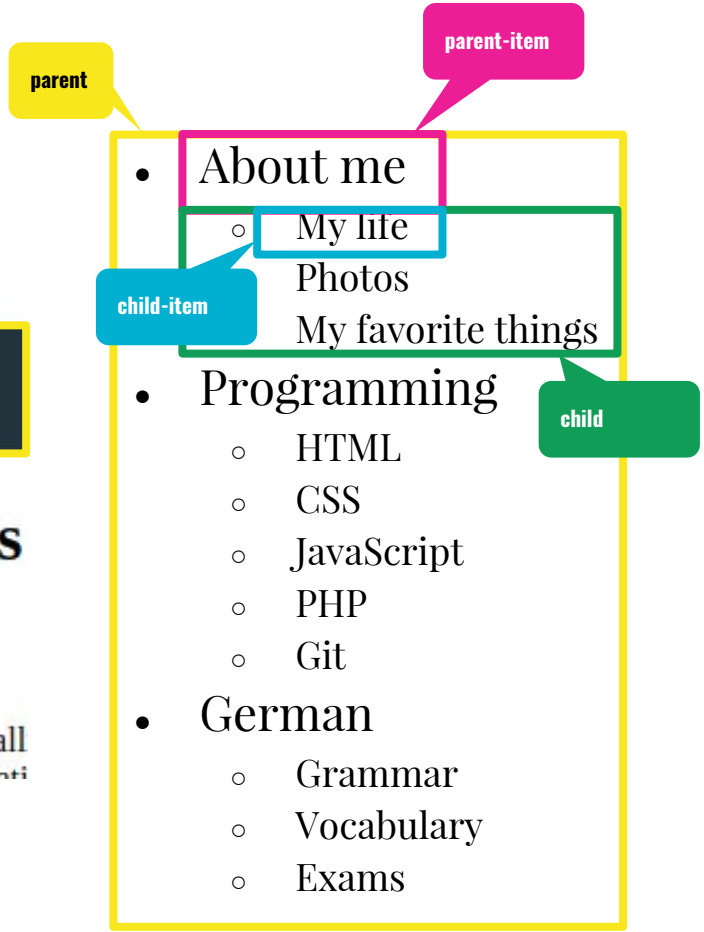

parent
parent-item
About me
Programming
German
My life
Photos
My favorite things
child
child-item
u using transitions
m
or sit amet, consectetur adipiscing elit. Praesent convall

Create list

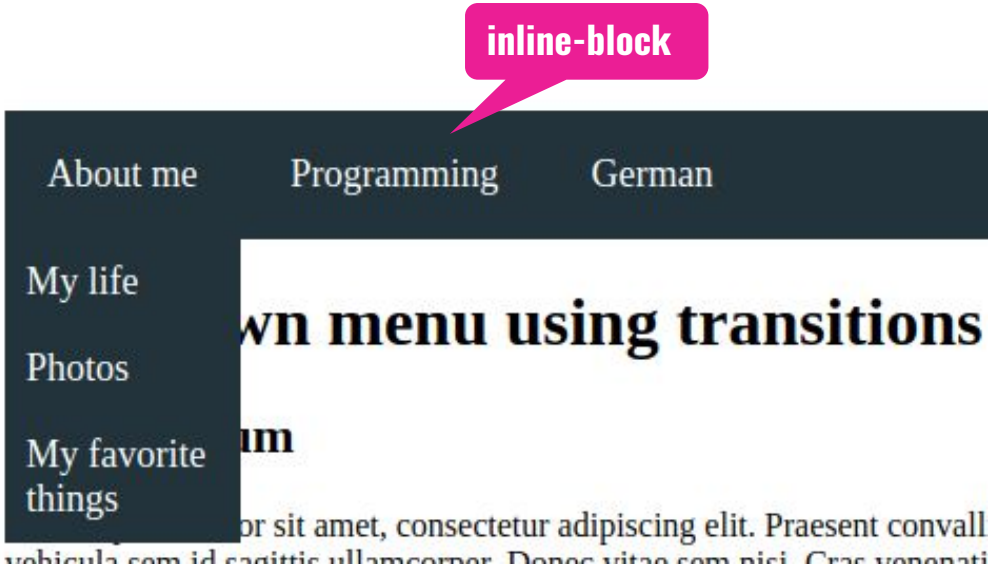
- About me
  - My life
  - Photos
  - My favorite things
- Programming
  - HTML
  - CSS
  - JavaScript
  - PHP
  - Git
- German
  - Grammar
  - Vocabulary
  - Exams

Assign class names

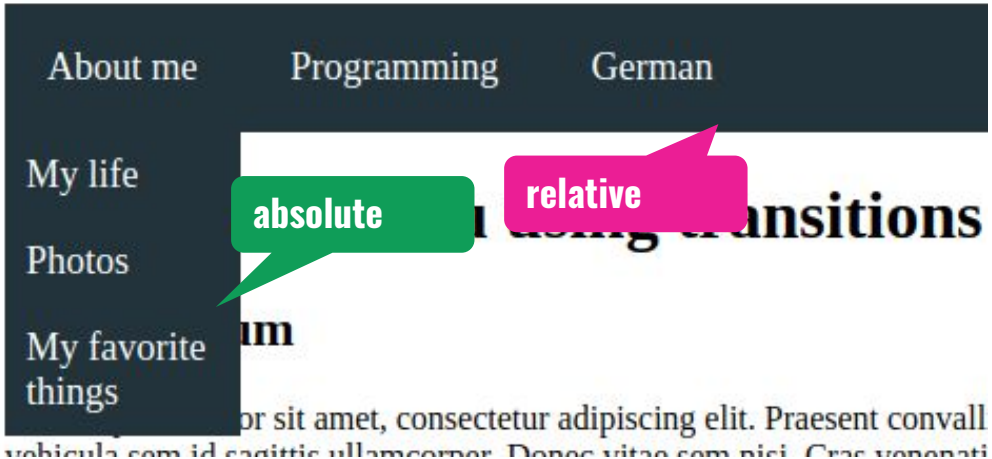




Make menu horizontally



Position parent-item and child



Hide the children

Transition!

When hovering the parent-item, the child should appear.

Make it extra nice

- Add `transform: translate(0, -30px)` to the child
- Change it `transform: translate(0)` when parent-item is hovered

Animations

Transitions vs. animations

Transitions:

- whenever a property changes, animate it
- don't loop
- require triggering (ex.: hover)
- no control between start and end

Animations:

- animation that plays on the element
- can loop
- don't require triggering
- you can define intermediate points (keyframes)

@keyframes

Includes the animation name, any animation breakpoints, and the properties intended to be animated.

```
@keyframes rotate {

  0% {
    transform: rotate(0deg);
  }

  100% {
    transform: rotate(360deg);
  }
}
```

@keyframes

You can have many intermediate points

```
@keyframes slide {

  0% {
    left: 0;
    top: 0;
  }

  50% {
    left: 244px;
    top: 100px;
  }

  100% {
    left: 488px;
    top: 0;
  }

}
```

animation-name

Defines which animation will be used

```
@keyframes slide {

     ...

}

#box {
     ...
     animation-name: slide;
}
```

animation-duration

How long will the animation last

```
@keyframes slide {

    ...

}

#box {
    ...
    animation-name: slide;
    animation-duration: 1s;
}
```

animation-iteration-count

How many times will the animation happen.

- `1` (default)
- `2, 3, …`
- `infinite`

```
@keyframes slide {

    ...

}

#box {
    ...
    animation-name: slide;
    animation-duration: 1s;
    animation-iteration-count: 10;
}
```

animation-direction

Defines if animation should play in reverse direction or in alternate cycles

- `normal` (default)
- `reverse`
- `alternate`
- `alternate-reverse`

```
@keyframes slide {

    ...

}

#box {
    ...
    animation-name: slide;
    animation-duration: 1s;
    animation-iteration-count: 10;
    animation-direction: alternate;
}
```

animation-timing-function

Specifies the speed curve of the animation

- `ease` (slow start, then fast, then end slowly, default)
- `linear` (same speed from start to end)
- `ease-in` (slow start)
- `ease-out` (slow end)
- `cubic-bezier(n,n,n,n)` (specify your own function)

```
@keyframes slide {

    ...

}

#box {
    ...
    animation-name: slide;
    animation-duration: 1s;
    animation-iteration-count: 10;
    animation-direction: alternate;
    animation-timing-function: linear;
}
```