Agenda

- **Tag 1 – Einführung**
  - Installation
  - Erste Anwendung
  - Architektur
- **Tag 2 – Entwicklung**
  - Models
  - Darstellung
  - Formulare
  - Ajax
- **Tag 3 – Fortgeschrittene Themen**
  - Tests
  - Security & Deployment
  - Lokalisierung & Internationalisierung
  - Performance
  - Best Practices

Ajax

Ajax

anderScore trust in competence

GFU Cyrus AG

JavaScript

- Erzeugung von JS für Komponenten durch Framework
- Einbindung von jQuery

```
JavaScriptLibrarySettings javaScriptSettings = getApplication().getJavaScriptLibrarySettings();

response.render(JavaScriptHeaderItem.forReference(javaScriptSettings.getJQueryReference()));
```

- Einbindung von eigenen Skripten
  - .js-Dateien als Resourcen
  - JavaScriptHeaderItems
  - OnDomReadyHeaderItems

Ajax

anderScore trust in competence

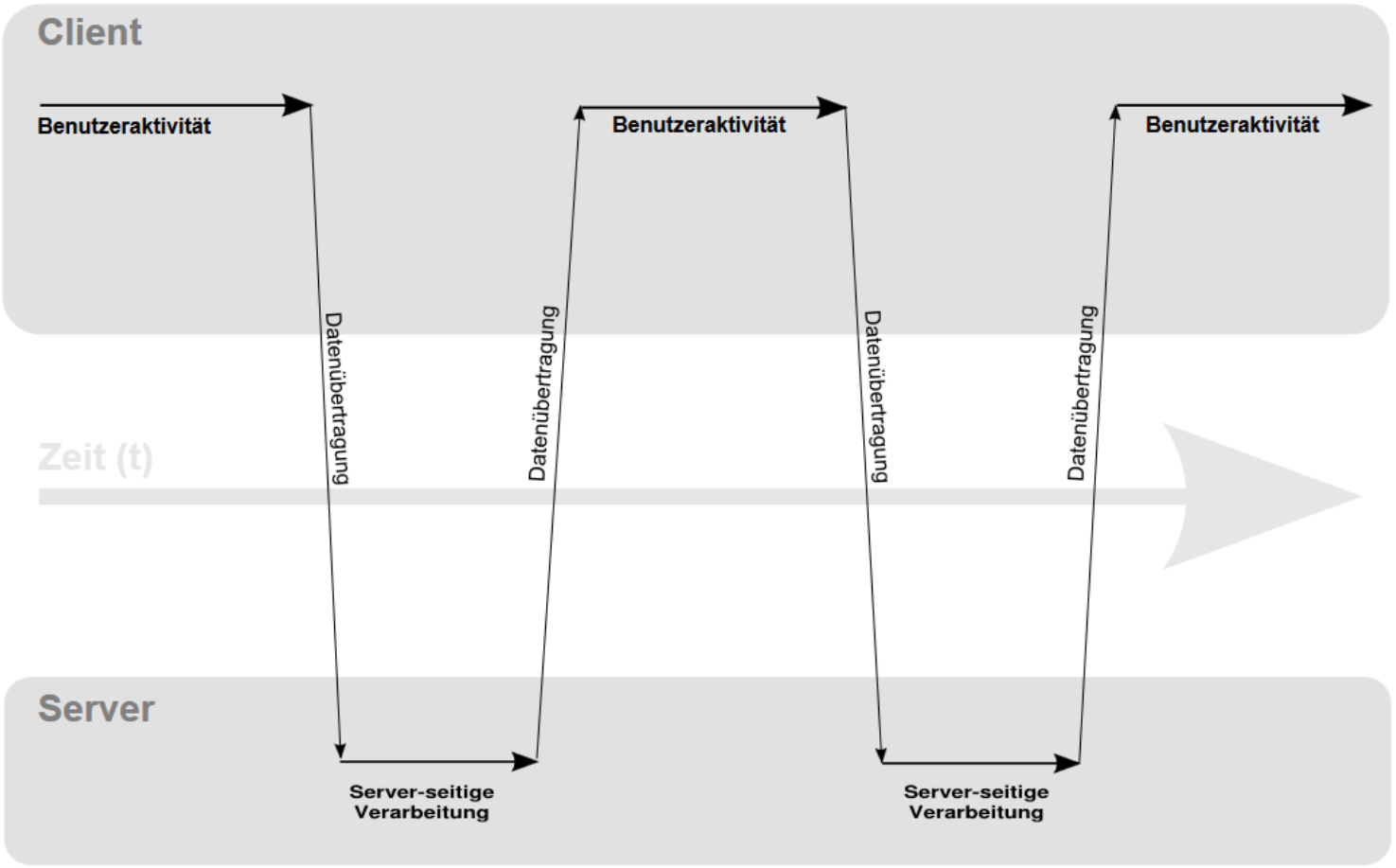
GFU Cyrus AG

Grundlagen

- AJAX: **A**synchronous **J**avaScript **a**nd **X**ML
- Funktionsweise
  - Kommunikation mit Server im Hintergrund (XMLHttpRequest)
  - Dynamische Manipulation des DOMs
  - Keine vollständigen Page Reloads
  - Keine Blockierung der UI
- Basis moderner, interaktiver Web-Anwendungen
- Historie
  - 1998: Microsoft ActiveX
  - 2005: Prägung des Begriffs durch Jesse James Garrett

Ajax

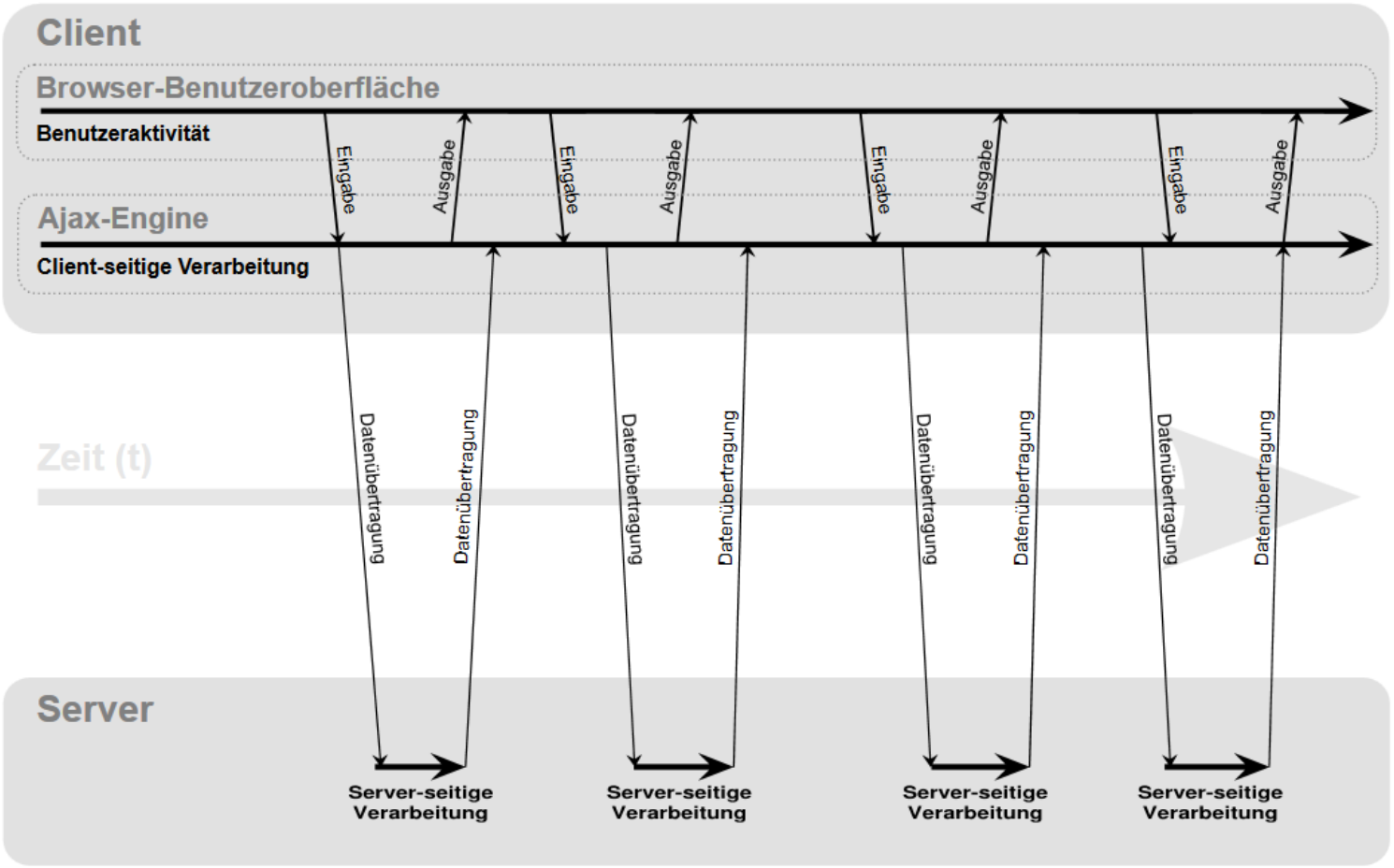
Klassisches Modell einer Web-Anwendung (synchrone Datenübertragung)



Quelle: Wikipedia (https://de.wikipedia.org/wiki/Ajax_(Programmierung)), 10.06.2016

Ajax

Ajax Modell einer Web-Anwendung (asynchrone Datenübertragung)



Quelle: Wikipedia (https://de.wikipedia.org/wiki/Ajax_(Programmierung)), 10.06.2016

Ajax

Grundlagen

- Eigene Ajax-Komponenten
  - Links, Buttons
  - Checkboxen
  - Editierbare Labels
  - Textfelder (inkl. autocomplete)
  - Modale Fenster
- Binden von AjaxBehaviours an Komponenten
- Interaktion mit *RequestTarget* in *Request Handlern*
  - Als Renderer zu verstehen
  - Angabe zu aktualisierender Komponenten
  - Repeater: Umgebenden Container angeben

Grundlagen

- Beispiel: Label

```
Label label = new Label("label", Model.of("Initial value."));
   // Komponente benötigt HTML ID, um aktualisiert werden zu können
   label.setOutputMarkupId(true);
   add(label);

   //…
    new AjaxLink("ajaxLink") {
        @Override
        public void onClick(Optional<AjaxRequestTarget> target) {
            if (target.isPresent()) {
                // Model Object verändern und Komponente neu rendern
                label.setDefaultModelObject("Another value 4 label.");
                target.get().add(label);
            }
        }
   };
```

Grundlagen

- Beispiel: Modales Fenster

```
<h2>Beispiel für ein Modal</h2>
<a wicket:id="openWindow">Öffnen</a>
<div wicket:id="modalWindow"></div>
```

Grundlagen

- Beispiel: Modales Fenster

```
public ModalPage(final PageParameters parameters) {
    super(parameters);

    ModalWindow modalWindow = new ModalWindow("modalWindow");
    Label label = new Label(modalWindow.getContentId(), Model.of("Hey!"));

    modalWindow.setContent(label);
    modalWindow.setTitle("Modal Window");

    add(modalWindow);
    add(new AjaxLink("openWindow") {
        @Override
        public void onClick(Optional<AjaxRequestTarget> target) {
            if (target.isPresent()) {
                modalWindow.show(target.get());
            }
        }
    });
}
```

Ajax

Ajax Behaviors

- Mittels Ajax auszuführende Aktionen
- Nachträgliches „Ajaxifizieren“ von Komponenten
- Vorgefertigte Behaviors
  - AjaxEventBehavior
  - AjaxFormSubmitBehavior
  - AbstractAjaxTimerBehavior

Ajax

AjaxEventBehavior

- Serverseitige Behandlung von JavaScript Events

- HTML

```
<div wicket:id="clickLabel"></div>
Sie haben <span wicket:id="counterLabel"></span> mal geklickt.
```

AjaxEventBehavior

```
public class AjaxPage extends WebPage {

    public AjaxPage(final PageParameters parameters) {
        super(parameters);

        ClickLabel clickLabel = new ClickLabel("clickLabel", Model.of("Bitte klicken!"));

        Label counterLabel = new Label("counterLabel", new PropertyModel(clickLabel, "counter"));
        counterLabel.setOutputMarkupId(true);

        clickLabel.add(new AjaxEventBehavior("click"){

            @Override
            protected void onEvent(Optional<AjaxRequestTarget> target) {
                if (target.isPresent()){
                    clickLabel.incrementCounter();
                    target.get().add(counterLabel);
                }
            }
        });

        add(clickLabel);
        add(counterLabel)
    }
}
```