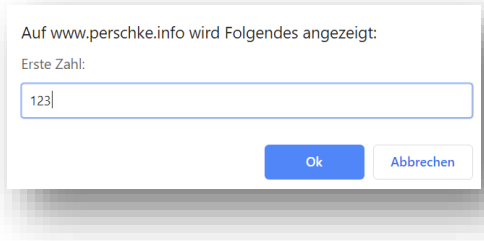
Möglichkeiten der Interaktion mit dem Anwender

In diesem Dokument werden einige Möglichkeiten zur Interaktion mit dem Anwender vorgestellt. Die Ausführungen sind möglichst einfach gehalten, auf Erläuterung der Hintergründe wurde verzichtet.

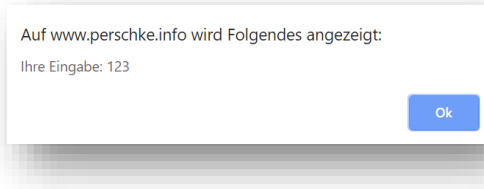
Einsatz modaler Dialoge

Modale Dialoge haben die Eigenschaft, dass der Programmablauf solange gestoppt wird, bis der Dialog beendet wird. Diese Art der Ein- / Ausgabe eignet sich immer dann, wenn nur wenige Ein- bzw. Ausgaben notwendig sind.

```
//Eingabe
var z1 = prompt("Erste Zahl: ");
```




```
//Ausgabe
alert("Ihre Eingabe: " + z1);
```



Die Funktion **prompt(…)** öffnet einen modalen Dialog mit dem als Paramter übergebenen Text (hier: „`Erste Zahl:` "). Zur Eingabe eines Wertes steht dem Anwender ein einzeiliges Texteingabefeld zur Verfügung. Nach Klick auf die Schaltfläche „Ok“ wird der Wert von der Funktion zurückgeliefert und kann beispielsweise einer Variablen zugewiesen werden.

Ausgaben können mit Hilfe der Funktion **alert(…)** realisiert werden. Der auszugebende Text muss der Funktion als Paramter mitgegeben werden.

Achtung:

- Die Funktionen prompt() und alert(…) blockieren die Programmausführung solange, bis der Anwender den Dialog mit „Ok“ oder „Abbrechen“ beendet.
- Bei dem von der Funktion prompt() gelieferten Wert handelt es sich immer um eine Zeichenkette. Möchte man mit dem vom Anwender eingegebenen Wert Berechnungen anstellen, so ist es ratsam, diesen zuerst in eine Zahl (numerischen Wert) zu wandeln. Hierzu können die Funktionen parseInt(…) oder parseDouble(…) verwendet werden. Dies könnte wie folgt aussehen:

  ```
  var z1 = parseInt( prompt("Erste Zahl: ") );
  ```

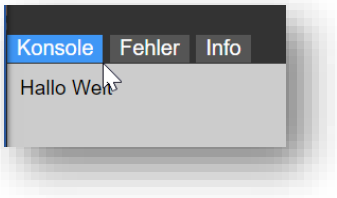
  Tipp: Alternativ kann man eine Zeichenkette auch durch die Multiplikation mit dem Faktor 1 in eine Zahl wandeln. Man nutzt hier die implizite Typenkonvertierung von Javascript. Beispiel:

  ```
  var z1 = 1 * Zahl1.value;
  ```

Ausgabe über die Konsole

Über die Anweisung `console.log(…)` können Ausgaben in die „Konsole“ erfolgen. In Thommy findet man diese Ausgaben im Reiter „Konsole“ innerhalb des Informationsbereichs am unteren Rand des Fensters.

```
//Ausgabe
console.log("Hallo Welt");
```

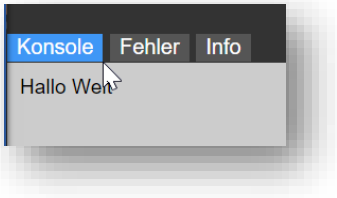


Achtung:
Die Anweisung `console.log(…)` eignet sich lediglich für die Ausgabe von Informationen.

Ausgabe über document.write

Ausgaben können auch über die Anweisung `document.write(…)` erfolgen. Der auszugebende Text ist als Parameter zu übergeben. Dieser erscheint, wie bei console.log(…) auch im Reiter „Konsole“ des Informationsbereichs (siehe oben).

```
//Ausgabe
document.write("Hallo Welt");
```



Achtung:
Die Anweisung `document.write(…)` eignet sich lediglich für die Ausgabe von Informationen.

„Automatisch generierte Formularfelder“

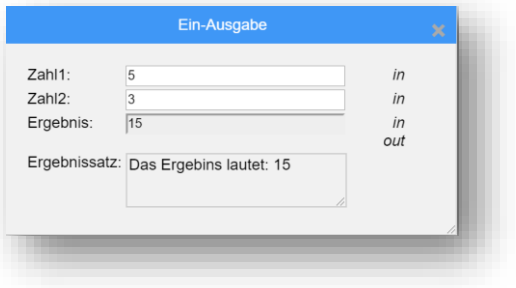
Werden mehrere Informationen vom Anwender benötigt, dann sind modale Dialoge eher unhandlich. Hier bietet sich eine kompakte Ansicht mit allen notwendigen Eingabefelder an. Die Thommy-IDE analysiert den Programmcode und kann die benötigten Felder erkennen und erzeugen. Diese werden im Ein- / Ausgabefenster dargestellt, welches über die Tastenkombination STRG+E aktiviert werden kann. Alternativ kann man die Sichtbarkeit auch über den Menüeintrag „Ansicht -> Ein-/Ausgabe-Fenster“ steuern. Das Fenster lässt sich in Position und Größe verändern. Voraussetzung ist, dass der Eintrag „GUI autom. generieren“ in der Rubrik „Ansicht“ aktiviert ist.

```
//Eingabe
var z1 = Zahl1.value;
var z2 = Zahl2.value;

var e = z1 * z2;


//Ausgabe
//  einzeilig
Ergebnis.value = e;

//  mehrzeilig
Ergebnissatz.innerText="Das Ergebnis lautet:" + e;
```



Ein- bzw. Ausgabefelder müssen einem speziellen Muster folgen:

```
NameDesFeldes.value
```

Der Name des Feldes muss dabei einfach durch `.value` ergänzt werden. Befindet sich der Ausdruck links von einer Zuweisung, dann handelt es sich um ein Ausgabefeld, ansonsten um ein Eingabefeld. Wählt man anstelle von `.value` die Ergänzung `.innerText`, dann wird anstelle eines einzeiligen, ein mehrzeiliger Ausgabebereich angezeigt. Der Bereich ist über das Ziehen der rechten unteren Ecke in der Größe veränderbar.

Achtung:

- Die Felder sind erst nach Aktivierung des Ein-/Ausgabe-Fenster sichtbar (STRG+E)
- Im Gegensatz zu modalen Dialogen wird hier nicht auf die Eingabe des Benutzers gewartet. Die Eingabe muss also vor dem Start des Programms erfolgen.
- Im Debug-Modus muss die Eingabe direkt nach dem Start erfolgen.
- Die Namen der Felder dürfen keine Leer- oder Sonderzeichen enthalten. Einzig erlaubtes Sonderzeichen ist der Unterstrich. Nutzen Sie diesen oder die CamelCase-Schreibweise, um übersichtliche, sprechende Namen zu erzeugen. Beispiel:

  ```
  Eingabe_Zahl1.value = …
  Das_Ergebnis_lautet.value = …
  DasErgebnisLautet.value = …
  ```

- Alternativ kann auch die Anweisung document.getElementById(…) genutzt werden. Beispiel:

  Statt

  ```
  Eingabe_Zahl1.value = …
  ```

  ist auch folgende, sehr viel kompliziertere Anweisung möglich

  ```
  document.getElementById("Eingabe_Zahl1").value = …
  ```

Automatisch generierte Schaltflächen (Buttons)

In einigen Fällen ist es vorteilhaft, wenn bestimmte Funktionalität erst nach Klick auf eine Schaltfläche (Button) ausgeführt wird. In der Regel werden Buttons bereitgestellt, die nach Klick eine vorgefertigte Funktion aufrufen. Auch hier kann die Thommy-IDE helfen. Voraussetzung dafür ist, dass der Eintrag „GUI autom. generieren“ in der Rubrik „Ansicht“ aktiviert ist.

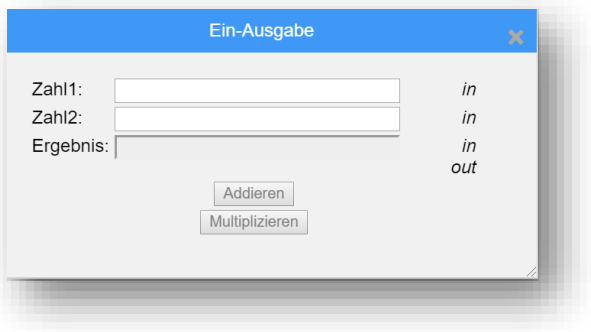
```
function sum() {
    var z1 = 1 * Zahl1.value;
    var z2 = 1 * Zahl2.value;

    Ergebnis.value = z1+z2;
}

function multi() {
    var z1 = 1 * Zahl1.value;
    var z2 = 1 * Zahl2.value;

    Ergebnis.value = z1*z2;
}


Addieren.onclick = sum;
Multiplizieren.onclick = multi;
```



Anklickbare Schaltflächen (Buttons) können über ein spezielles Muster erstellt werden:

```
NameDesButtons.onclick = NameDerFunktion;
```

Eine solche Zeile erstellt im Ein-/Ausgabefenster eine zusätzliche Schaltfläche (Button), nach dessen Klick die angegebene Funktion aufgerufen wird.

Die im Beispielcode zu findende Zeile `Addieren.onclick = sum;` erzeugt eine Schaltfläche mit der Aufschrift „Addieren“. Nach Klick auf diese Schaltfläche wird die Funktion `sum` aufgerufen.

<u>Achtung:</u>

- Die Schaltflächen sind erst nach Aktivierung des Ein-/Ausgabe-Fenster sichtbar (STRG+E)
- Standardmäßig sind die Schaltflächen deaktiviert. Erste nach Programmstart (STRG-F11) werden diese aktiviert.
- Die Namen der Schaltflächen dürfen keine Leer- oder Sonderzeichen enthalten. Einzig erlaubtes Sonderzeichen ist der Unterstrich.
- Bei der Zuweisung dürfen hinter dem Namen der Funktion keine runden Klammern vorhanden sein!
- Erfolgt der Aufruf einer Funktion über eine Schaltfläche, wird dies im Debug-Modus leider nicht erkannt. Möchte man eine Funktion debuggen, dann muss der Aufruf also manuell erfolgen.

- Möchte man einer Funktion nach Klick auf eine Schaltfläche Parameter mitliefern, verkompliziert sich die Struktur erheblich:

  ```
  Addieren.onclick = function() { sum(5, 12) };
  ```

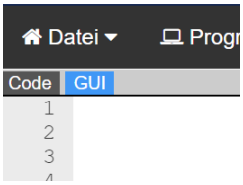
  Der eigentliche Aufruf (hier: `sum(5, 12)`) muss dann in eine anonyme Funktion eingepackt werden.

  In der Regel kann man die Übergabe von Parametern durch zusätzliche Eingabefelder vermeiden.

Erstellen einer benutzerdefinierten GUI

Thommy kann Ihren Programmcode analysieren und automatisch eine GUI mit allen notwendigen Ein- / Ausgabefeldern und Schaltflächen erzeugen (siehe voriger Abschnitt). Alternativ können Sie auch selbst eine GUI erstellen. Deaktivieren Sie dazu den Eintrag „GUI autom. generieren“ in der Rubrik „Ansicht“.

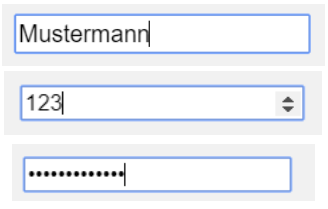
Oberhalb des Editorbereichs finden Sie einen Reiter „GUI“. Sie können hier alle HTML-Anweisungen ablegen, die zur Darstellung der GUI notwendig sind. CSS-Anweisungen können entweder inline (style-Element) oder intern (style-Attribut) eingebunden werden. Es lassen sich so sehr aufwendige und schicke GUIs erstellen.

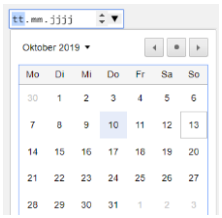


In der Regel wird die GUI über Ein- und Ausgabefelder verfügen. Ihnen stehen dabei alle Möglichkeiten zur Verfügung, die auch in einer regulären HTLM-Seite denkbar sind. Im folgenden Abschnitt finden Sie eine Aufstellung mit verschiedenen GUI-Elementen, deren Umsetzung in HTML und das Ansprechen mittels Javascript.

Anmerkung: Neben den hier vorgestellten Ein- / Ausgabeelementen können Sie natürlich auch beliebigen zusätzliche HTML-Elemente einbinden, beispielsweise eine Überschrift (<h1>).

Einzeiliges Ein- / Ausgabefeld, Passwortfelder, …





HTML (GUI)

```
<input type="text" id="nachname">
```

Javascript (Code)

In Feld schreiben:

```
document.getElementById("nachname").value = "Mustermann"
```

oder einfacher

```
nachname.value = "Mustermann"
```

Aus Feld lesen:

```
var n = document.getElementById("nachname").value
```

oder einfacher

```
var n = nachname.value;
```

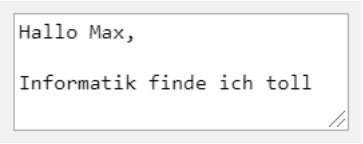
Bemerkungen:

- Die „id“ (hier: nachname) dient für den Zugriff auf das Element. Sie können diese frei wählen.
- Sie können den Typ des Eingabefeldes variieren. Eine Übersicht finden Sie beispielsweise auf der Seite https://www.w3schools.com/html/html_form_input_types.asp.

Mehrzeilige Ein- / Ausgabefeld

HTML (GUI)

```
<textarea id="bemerkung"></textarea>
```



Javascript (Code)

In Feld schreiben:

```
document.getElementById("bemerkung").value = "Hallo Max,\nInformatik finde ich toll"
```

oder einfacher

```
bemerkung.value = "Hallo Max,\nInformatik finde ich toll"
```

Aus Feld lesen:

```
var b = document.getElementById("bemerkung").value
```

oder einfacher

```
var b = bemerkung.value;
```

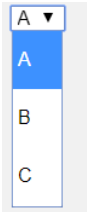
Bemerkungen:

- Die „id“ (hier: `bemerkung`) dient für den Zugriff auf das Element. Sie können diese frei wählen.
- Mit der Zeichenkombination „\n“ können Sie innerhalb des Texteingabefeldes einen Zeilenumbruch setzen

Listen

HTML (GUI)

```
<select id="wert">
    <optione>A</option>
    <optione>B</option>
    <optione>C</option>
</select>
```



Javascript (Code)

In Feld schreiben:

```
document.getElementById("wert").value = "A"
```

oder einfacher

```
wert.value = "A"
```

Aus Feld lesen:

```
var w = document.getElementById("wert").value
```

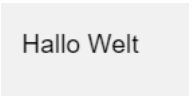
oder einfacher

```
var w = wert.value;
```

Bemerkungen:

- Die „id“ (hier: `wert`) dient für den Zugriff auf das Element. Sie können diese frei wählen.
- Ergänzen Sie den Starttag um das Attribut „size“ mit einem Wert größer 0, um eine echte Liste zu erhalten. Beispiel:
  `<select id="wert" size="3">  …`

Nichtbeschreibbarer Ausgabebereich



HTML (GUI)

```
<p id="ausgabe"></p>
```

Javascript (Code)

In Feld schreiben:

```
document.getElementById("ausgabe").innerText = "Hallo Welt"
```

oder einfacher

```
ausgabe.value = "Hallo Welt"
```

Aus Feld lesen:

```
var b = document.getElementById("ausgabe").innerText
```

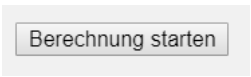
oder einfacher

```
var b = ausgabe.innerText;
```

Bemerkungen:

- Die „id“ (hier: `ausgabe`) dient für den Zugriff auf das Element. Sie können diese frei wählen.
- Zeilenumbrüche können Sie im Ausgabebereich mit der HTML-Tag `<br>` erstellen.
- Anstelle eines p-Elements können Sie auch ein fast beliebiges anderes Element nutzen.
- Ersetzen Sie im Beispiel „innerText“ durch „innerHTML“, um HTML-formatierten Text darzustellen.

Schaltflächen



HTML (GUI)

```
<input type="button" value="Berechnung starten" onclick="berechnen()">
```

oder

```
<button onclick="berechnen()">Berechnung starten</button>
```

Javascript (Code)

```
function berechnen() {
    …
}
```

Bemerkungen:

- In der Regel nutzen Sie Schaltflächen, um bei Klick eine Javascript-Funktion aufzurufen.
- Das HTML-Attribut „onclick“ nennt sich auch Eventhandler. Die Funktion wird in diesem Fall aufgerufen, sobald die Schaltfläche geklickt wurde. Es gibt zahlreiche weiterer Eventhandler. Um mehr zu erfahren, besuchen Sie die Seite https://www.w3schools.com/js/js_events.asp.
- Prinzipiell lassen sich solche Eventhandler auch mit vielen anderen HTML-Elementen kombinieren. Denkbar ist beispielsweise ein Bild, welches bei Klick eine Funktion aufruft:

```
<img src="…" onclick="berechnen()">
```