



# Girls Who Code At Home

Virtuelle Wanderung  
Interaktive Web-App mit JavaScript

## Übersicht über die Aktivität

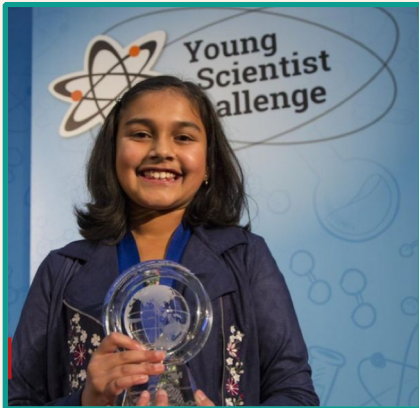
Der Tag der Erde steht diese Woche bevor. Daher möchten wir dich anregen, an Plätze in der freien Natur zu denken, die du gerne entdecken möchtest. Wo würdest du gerne hingehen? Was würdest du unternehmen und sehen wollen? In diesem Tutorial lernst du, wie du **Arrays** oder Listen in JavaScript verwendest, um eine virtuelle Wandung zu konfigurieren. Du erstellst eine Website, die andere mit auf eine Reise nehmen kann – in einen Nationalpark, dein Lieblingsmuseum oder ein fremdes Land. Die Möglichkeiten sind unbegrenzt. Bevor du an deiner Reiseroute zu arbeiten beginnst, lerne die in „Frauen im Rampenlicht der Technik“ vorgestellte Gitanjali Rao kennen.

**Hinweis:** Bei der Erstellung einer Website bauen viele Programmierer mit HTML das Gerüst, mit CSS gestalten sie die Site und mit JavaScript wird sie interaktiv. In dieser Aktivität befassen wir uns hauptsächlich mit JavaScript. Wenn du eine schnelle Auffrischung für HTML und CSS haben möchtest, schau dir unbedingt unsere dritte Aktivität an: [Teile dein Können](#), bevor du mit dieser Aktivität beginnst.

## Materialien

- [Glitch](#) oder dein Lieblings-Texteditor
- [Beispielprojekt virtuelle Wanderung \(mit Erweiterungen\)](#)
- [Beispielprojekt virtuelle Wanderung \(ohne Erweiterungen\)](#)
- [Startercode Virtuelle Wanderung](#)
- Wahlweise: Planungshilfe
- Wahlweise: Kugelschreiber/Bleistifte/Marker

## Frauen im Rampenlicht der Technik: Gitanjali Rao



**Bildquelle:** [BBC](#)

Im Jahr 2014 gab es in Flint in Michigan ein öffentliches Gesundheitsproblem. Das Wassersystem von Flint enthielt Blei. Dies ist ein extrem gefährlicher Stoff, wenn er verzehrt wird – insbesondere für Kinder. Als Gitanjali von diesem Gesundheitsproblem erfuhr, handelte sie und erfand Tethys, einen Sensor, der Verunreinigung im Wasser aufspürt.

Gitanjali war gerade mal 11 Jahre alt, als sie Tethys kreierte. Sie gewann 2017 den [„Discovery Education 3M Young Scientist Challenge“-Jugendwettbewerb](#) und wurde als Rednerin zu TEDx eingeladen.

Ihr Sensor basiert auf einem Kohlenstoff-Nanoröhrchen, das den Fluss von Elektronen in Blei aufspürt, aber auch auf ein Arduino-basiertes Signal zugreift, das über Bluetooth mit einer Smartphone-App verknüpft wird.

Schau dir das [Video](#) über Gitanjali Rao an und erfahre mehr über ihre Erfindung Tethys. Wenn du Zeit hast, empfehlen wir dir, mit [diesem Artikel](#) von Bustle, [diesem](#) vom Young Scientist Lab oder [diesem](#) von NPR mehr über Gitanjali zu erfahren. Du kannst sie auch bei TEDx-Talk [hier](#) sprechen hören.

## Überlegung

Informatikerin zu sein, bedeutet wesentlich mehr, als nur gut programmieren zu können. Nimm dir etwas Zeit und denke darüber nach, inwiefern Gitanjali und ihre Arbeit jene Stärken ausdrücken, um die sich bedeutende IT-Wissenschaftler\*innen bemühen – Mut, Hartnäckigkeit, Kreativität und sinnvolle Ziele.



**ZIEL**

Gitanjali interessiert sich für den Medizinbereich, aber lernte dennoch zu programmieren und Technologie zu konstruieren. Für welche anderen Bereiche interessierst du dich, für die du Informatik verwenden könntest?

Bespreche deine Ergebnisse mit einem Familienmitglied oder im Freundeskreis. Ermutige andere, mehr über Gitanjali zu lesen und sich am Gespräch zu beteiligen!

## Schritt 1: Erkundungen (5 Min.)

In diesem Tutorial lernst du, eine Website mit einer Reihe von Bildern zu erstellen, die den Benutzer auf eine virtuelle Wanderung mitnehmen. Nimm dir 5 Minuten Zeit, um dir die Funktionen in dieser [Beispiel-Website](#) anzuschauen, die wir mit Bildern von Plätzen in der Natur Japans erstellt haben.

- **Thema des Projekts:** Natur
- **Publikum:** Menschen, die mehr über Plätze in der Natur Japans erfahren möchten
- **Ziel:** Ein Erlebnis im Freien teilen und eine Reiseroute durch die Natur Japans planen

Stelle dir beim Erkunden der Website folgende Fragen:

- Was passiert, wenn du auf die einzelnen Schaltflächen klickst?
- Haben die ausgewählten Fotos einen Bezug zum Thema und Ziel? Welche Fotos würdest du gerne verwenden?
- Welche Schritte erwartest du, wenn du auf die Schaltfläche „Weiter“ klickst? Versuche die Schritte in noch kleinere Etappen aufzuschlüsseln.
- Wann funktioniert die Schaltfläche „Weiter“ nicht mehr? Wann funktioniert die Schaltfläche „Zurück“ nicht mehr? Warum könnte dies der Fall sein?

## Schritt 2: Mache ein Brainstorming zu Funktionen und plane dein Projekt (10 Min.)

Nachdem du das Beispielprojekt erkundet hast, könntest du dir etwas Zeit nehmen, um zunächst deine Strategie zu definieren. Nutze diese Zeit, um dir klarzuwerden, was dein Projekt leisten soll und was dein Ziel ist. Du kannst ggf. das Planungsdokument am Ende dieser Aktivität als Hilfestellung verwenden, um deine Ideen zu festzuhalten und zu ordnen.

### 1. Wähle ein Thema, ein Publikum und ein Ziel für deine virtuelle Wanderung.

Zu Ehren des Tags der Erde (22. April 2020) möchten wir dich anregen, einen Platz im Freien auszuwählen, den du gerne entdecken möchtest. Du kannst aber auch jedes andere Thema wählen. Falls du nicht weiterkommst, findest du hier mögliche Projektideen:

- Eine virtuelle Galerie mit Aktivitäten/Hobbys, die du zu Hause gerne machst.
- Ein virtueller Spaziergang durch dein Lieblingsmuseum.
- Eine virtuelle Reise durch deine Lieblingsorte.

Wenn du das Publikum für deine virtuelle Wanderung definierst, denke über deine Zielgruppe nach. Wer könnte deine Website spannend finden? Berücksichtige dabei, was deine Zielgruppe gerne erfahren möchte und wie du ihre Aufmerksamkeit wecken wirst.

### 2. Stelle mindestens drei Bilder zusammen.

Du kannst mindestens drei Bilder verwenden, die du auf deiner virtuellen Wanderung zeigen kannst. Du kannst eigene Bilder auswählen oder Bilder, die du online findest. Wenn du deine Bilder zusammenstellst, denke auch an den Titel oder den kurzen Werbetext, den du zu jedem Bild zeigen möchtest.

## Schritt 3: Erste Schritte mit Glitch (10 Min.)

Glitch ist ein einfaches Tool zur Erstellung von Web-Apps. Es enthält einen Texteditor, der dir die Änderungen an deiner Website in Echtzeit anzeigt. Außerdem kannst du damit dein Projekt ganz einfach veröffentlichen, sodass die ganze Welt es sehen kann! Wenn du bei jemand anderem ein cooles Projekt siehst, kannst du dir dessen Code anschauen und ihn neu zusammenstellen.

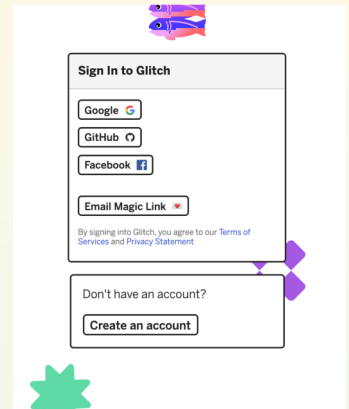
1. **Melde dich mit deinem Google-, Facebook- oder GitHub-Konto bei Glitch an.**

Um deine Arbeit in Glitch mit anderen teilen zu können, musst du angemeldet sein. Du kannst dich bei Glitch mit deinem Google-, Facebook- oder GitHub-Konto anmelden. Wenn du unter 13 Jahre alt bist, brauchst du für die Anmeldung die Zustimmung und die E-Mail-Adresse deiner Eltern.

2. **Erstelle diesen [Startercode](#) für deine Website mit der virtuellen Wanderung neu.** Schau dir dieses [Video](#) für detailliertere Anleitungen an.
3. **Erkunde die Oberfläche von Glitch.**

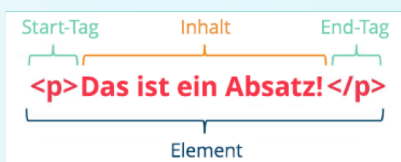
Als Erstes siehst du die Inhalte der Datei **README.md**. README-Dateien sind Dokumente, die dir in Projekten von anderen immer begegnen. Sie enthalten üblicherweise Informationen, wie man die Projektdateien durchsuchen und das Programm ausführen kann. Schau dir dieses [Video](#) an, um mehr über die Glitch-Benutzeroberfläche zu erfahren.

4. **Stelle deine Projektansicht ein.** Klicke auf die Schaltfläche **Anzeigen** und wähle die Option **Weiter zum Code** aus. Damit kannst du deinen Code und deine Projektansicht im Web nebeneinander sehen. Schau dir dieses [Video](#) für detailliertere Anleitungen an.



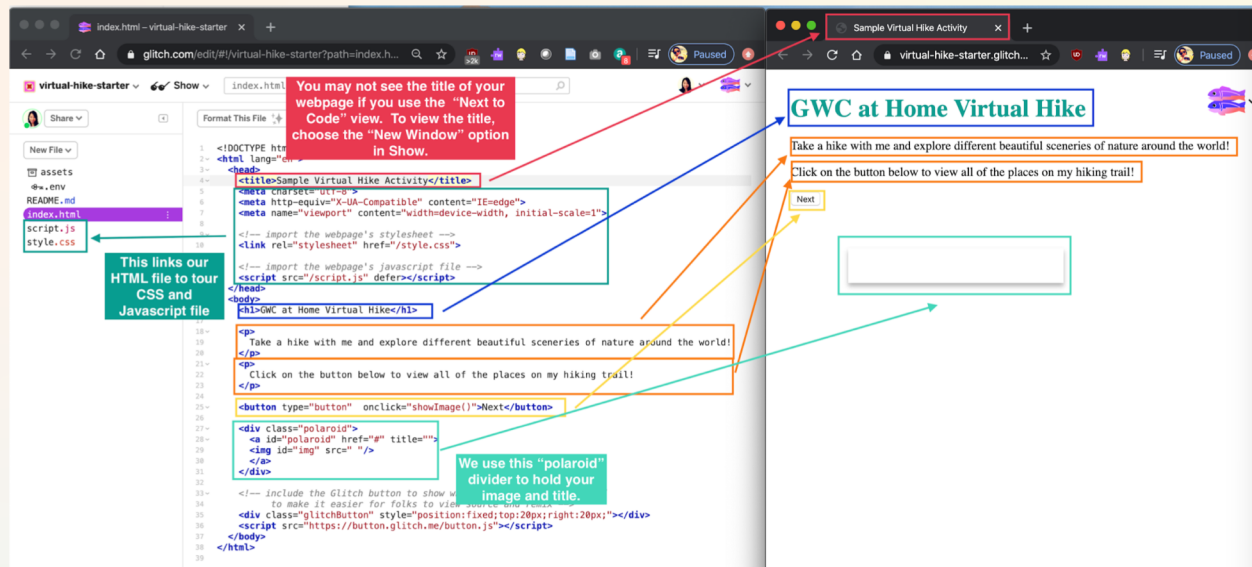
## Schritt 4: Erkunde das HTML-Startprojekt (10 Min.)

Schauen wir uns zuerst die Datei **index.html** an. Klicke im linken Navigationsmenü auf die Datei **index.html**. **HTML**, die Abkürzung von Hypertext Markup Language, verwendet **Tags**, um Inhalte für eine Website zu organisieren. Alle Inhalte in deiner Livevorschau sind von einem öffnenden Tag **<body>** und einem schließenden Tag **</body>** umschlossen. Tags sind ein Bestandteil der Anatomie eines HTML-Elements, das üblicherweise aus einem öffnenden Tag, aus Inhalt und einem schließenden Tag besteht. Auf den Bildern weiter unten siehst du Beispiele für HTML-Tags für verschiedene Elementtypen. Das Tag **<p>** wird verwendet, um „paragraph“ (Abschnitt) zu kennzeichnen, während das Tag **<a>** eher ein generisches Tag ist, das üblicherweise Links zugewiesen ist.



## Schritt 4: Erkunde das HTML-Startprojekt (Fortsetzung)

Das untenstehende Bild ordnet die Tags in deinem Startercode dem zu, wie sie in deiner Website angezeigt werden.



Nehmen wir uns etwas Zeit, um die Information für *deine* Website anzupassen.

### Versuche selbst HTML zu programmieren!

- 1. Ändere den Titel deiner Website.** Der Titel deiner Website entspricht dem Namen, der angezeigt wird, wenn du die Website lädst. Beachte bitte, dass dies nicht das gleiche ist, wie die URL deiner Website zu ändern. Wenn du prüfen möchtest, ob der Titel deiner Website richtig ist, klicke auf **Show** (Anzeigen) und wähle im Menü die Option **New Window** (Neues Fenster) aus.
  - Suche in der HTML-Datei das Tag `<title>`. (Eingrahmt in das rote Kästchen im Bild oben).
  - Ersetze den Text zwischen dem öffnenden Tag `<title>` und dem schließenden Tag `</title>` durch den Namen deiner Website.  
Bsp. `<title>Name meiner neuen Website</title>`
- 2. Ändere die Kopfzeile deiner Website.** Eine Kopfzeile ist der Text, der üblicherweise oben auf deiner Seite erscheint. Im Gegensatz zum Titel deiner Website ist die Kopfzeile in deiner Seite zu sehen.
  - Suche in der HTML-Datei das Tag `<h1>`. (Eingrahmt in das blaue Kästchen im Bild oben).
  - Ersetze den Text zwischen dem öffnenden Tag `<h1>` und dem schließenden Tag `</h1>` durch den Titel deines Projekts. Das könnte auch dem gleichen, was du als Titel für deine Website festgelegt hast.  
Bsp. `<h1>Name meiner neuen Website</h1>`

## Schritt 4: Erkunde ein HTML-Startprojekt (Fortsetzung)

3. **Ändere den Text auf deiner Website.** Im Beispielcode gibt es zwei Abschnitte. Der erste enthält eine kurze Beschreibung des Projekts, während der zweite dem Benutzer Anweisungen bereitstellt, wie er mit der Website interagieren kann.

- Suche das Tag `<p>` in der HTML-Datei. (Eingerahmt vom orangefarbenen Kästchen im Bild oben)
- Ersetze den Text zwischen dem **ersten Set** des öffnenden Tags `<p>` und des schließenden Tags `</p>` durch eine kurze Beschreibung deines Projekts. Lies noch mal nach, was du in deinem Planungsdokument geschrieben hast.

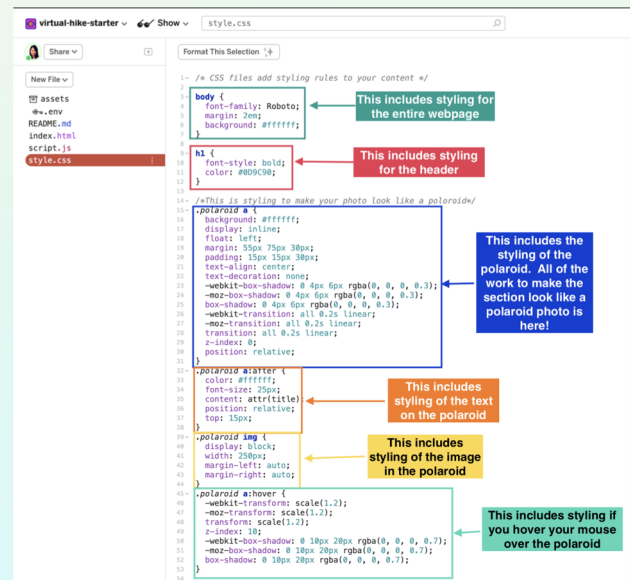
Bsp. `<p>Textbeschreibung Meine neue Website</p>`

- **(Optional)** Passe die Anweisungen im zweiten Set der `<p>`-Tags an.

## Schritt 5: Erkunde das Startprojekt CSS (5 Min.)

**CSS**, Abkürzung für **Cascading Style Sheets**, beschreibt die Gestaltungsvorlagen (oder Stylesheets), die in den HTML-Elementen angewendet werden sollen. CSS ermöglicht dir, das Erscheinungsbild der Inhalte deiner Website zu verändern, einschließlich Textfarbe, Textgröße, verwendete Schriften, Hintergrundfarben oder Bilder und vieles mehr. In diesem Tutorial gehen wir auf CSS nicht näher ein. Wir möchten dich anregen, dir [diese](#) Ressource und unsere Aktivität [Teile dein Können](#) anzuschauen, wenn du mehr über CSS erfahren möchtest.

In diesem Startprojekt sind bereits viele grundlegende CSS-Stylesheets enthalten, sodass du dich darauf konzentrieren kannst, JavaScript anzupassen und zu erlernen. Wir haben Stylesheets für deinen Titel, deine Kopfzeile und für Polaroid eingefügt. Wir haben eines der Elemente **polaroid** genannt, denn wir haben es in CSS-Stylesheets hinzugefügt, damit deine Fotos wie Polaroid-Aufnahmen aussehen. Stylesheets für das Polaroid wurden aus [diesem](#) Tutorial entliehen. Am Ende dieses Projekts hast du auch Zeit, deine Website in Bezug auf Farben und Schriften nach deinem Geschmack zu gestalten.



Bildquelle: [Creative Market](#)

## Schritt 6: Einführung in JavaScript (2 Min.)

Und nun kommt der lustige Teil! Du hast bisher das Gerüst für deine Website erstellt, aber noch tut sie nichts. Das ist der Moment, an dem **JavaScript** ins Spiel kommt. JavaScript wird verwendet, um Websites interaktiv zu machen. Schauen wir uns die Datei `script.js` an. Damit du einen leichten Einstieg findest, haben wir Dinge hinzugefügt, auch einige Kommentare als Erklärung, was jede Codezeile tut.

```
1- /* If you're feeling fancy you can add interactivity
2-    to your site with Javascript */
3-
4- //This is an array that will hold the file names! Add at least 4 images to the images array.
5- var images = new Array();
6- images[0] = "";
7- //This is an array that holds the names of each location shown in your pictures.
8- //Make sure that the index of each location is the same as the index of the picture in the images array
9- var locations = new Array();
10- locations[0] = "";
11-
12- //This is your starting index. First we start off at Home!
13- var index = 0;
14-
15- //This function will help us toggle through our photos. Everytime the button is pressed it will call this function!
16- function showImage() {
17-
18- }
19-
```

**Hinweis:** Kommentare werden vom Computer nicht gelesen – sie beschreiben den Code für andere oder für dich zu einem späteren Zeitpunkt. Du kannst zu einer Zeile einen Kommentar hinzufügen, indem du `//` und eine Nachricht einfügst.

## Schritt 7: Einführung in Variablen und Arrays (2 Min.)

Um Informationen zu speichern (wie unsere Fotos und deren Beschreibungen), benötigen wir eine **Variable**. Eine Variable ist ein Container, der alle Datentypen speichert. Sie können Wörter, Zahlen oder sogar eine Datenliste enthalten. Schauen wir uns die wichtigen Symbole (oder die Syntax) zur Verwendung von Variablen an:

`var myVariable = value;`

- **var:** Dieses Schlüsselwort wird verwendet, um einen variablen Typ zu bezeichnen.
- **myVariable:** Das ist der Name unserer Variablen. Namen werden üblicherweise als [CamelCase-Notation](#) dargestellt – ohne Leerstellen und mit einem Großbuchstaben am Anfang eines jedes neuen Worts.
- **=:** Das Gleichheitszeichen zeigt die Zuweisung oder Wiederzuweisung eines Wertes zu der Variable
- **value:** Das kann jeder Wert sein – eine Zahl, Wörter oder sogar eine Datenliste.
- **;;** Alle JavaScript-Anweisungen müssen mit einem Strichpunkt enden. So weiß der Computer, dass der Befehl beendet ist.

Ein **Array** ist eine geordnete Datenstruktur, die mehrere Informationstypen enthalten kann. Du kannst dir ein Array wie eine Kommode mit vielen Schubladen vorstellen, von denen jede Daten enthalten kann. Lass uns die wichtigen Symbole aufschlüsseln (oder die Syntax), die verwendet werden, um ein **leeres** Array (wie im Startercode) zu erstellen:

`var myArray = new Array();`

- **var:** Wir verwenden das Schlüsselwort **var**, um das Array in einer Variablen zu speichern.
- **new:** Dieses Schlüsselwort wird verwendet, um zu zeigen, dass wir ein leeres Objekt erstellen wollen. In unserem Fall ein leeres Array.
- **Array:** Das Schlüsselwort gibt dem Computer den Befehl, ein Array-Objekt zu erstellen.
- **():** Die Klammern werden verwendet, um die Funktion **Array** aufzurufen. JavaScript hat bereits eine integrierte Funktion (oder ein Set von Befehlen), die hilft, ein Array-Objekt zu erstellen. Wir werden in diesem Projekt zu einem späteren Zeitpunkt mehr über Funktionen und ihre Verwendung erfahren.

## Schritt 8: Verwendungsweise von Arrays (10 Min.)

Arrays gehören zu den meist verwendeten Datenstrukturen, denn sie haben eine *geordnete* Datenstruktur. Ein Array weist jedem Element, das es speichert, eine Zahl zu; diese Zahl bezeichnen wir als Index. Jedem Element einen Indexwert zuzuweisen, erleichtert das Zugreifen, Löschen und Ersetzen von Werten, die in diesem Array gespeichert sind. In der Informatik beginnen wir mit der Indexnummerierung bei 0. Das erste Element in dem Array wird dem Index 0 zugewiesen und dieser Index erhöht sich mit jedem nachfolgenden Element um 1. Schauen wir uns ein Beispiel an:

```
var programmers = ["Ada", "Grace", "Katherine", "Roya"];
```

In diesem Beispiel haben wir ein Array namens `programmers`, das den Namen von berühmten Programmierern enthält: `"Ada"`, `"Grace"`, `"Katherine"`, `"Roya"`. Insgesamt hat dieses Array eine Länge von 4. Die eckigen Klammern `[]` werden verwendet, um den Beginn und das Ende der Array-Inhalte anzuzeigen.

0	1	2	3
<code>"Ada"</code>	<code>"Grace"</code>	<code>"Katherine"</code>	<code>"Roya"</code>

Stellen wir uns vor, dass wir auf den Namen „Grace“ in dem Array zugreifen wollen. Wie würden wir das tun?

Wir können einfach den mit „Grace“ verbundenen Index verwenden, um diese Information abzurufen. Wir würden einfach die Anweisung: `programmers[1]` schreiben und somit zu dem Namen „Grace“ zurückkehren. Als Erstes geben wir (1) den **Namen** des Arrays ein, auf das wir zugreifen möchten und (2) fügen eckige Klammern hinzu `[]`, dann (3) geben wir die Indexnummer ein, auf die wir in der eckigen Klammer zugreifen wollen. Und da wir den Namen „Grace“ haben wollen, geben wir die Nummer 1 ein.

Gäben wir `programmers[3]` ein, würde dies den Namen „Roya“ ergeben. Wenn wir `programmers[4]` eingeben, würden wir einen **error** (Fehler) bekommen! Arrays werden mit **0 indexiert**, das bedeutet, dass der Index bei 0 beginnt. Selbst, wenn es in der Liste 4 Namen gibt, hat der Name am Ende einen Index, der um *einen Wert kürzer ist als die Länge*.

Wir können den Index auch verwenden, um zum Array einen Wert hinzuzufügen oder darin einen Wert zu ändern. Stellen wir uns vor, wir wollen den Namen „Gitanjali“ zu unserem Array hinzufügen. Wir können die folgende Syntax eintippen:

```
programmers[4] = "Gitanjali";
```

## Schritt 9: Füge dein erstes Bild und den ersten Standort hinzu (10 Min.)

Schau dir noch mal die Datei `script.js` an. Beachte, dass wir die zuvor gezeigte Syntax verwenden, bevor wir zwei verschiedene Arrays erstellen, eines mit dem Namen `images` und eines mit dem Namen `locations`.

```
/* If you're feeling fancy you can add interactivity
   to your site with Javascript */

//This is an array that will hold the file names! Add at least 4 images to the images array.
var images = new Array();
images[0] = "";
//This is an array that holds the names of each location shown in your pictures.
//Make sure that the index of each location is the same as the index of the picture in the images array
var locations = new Array();
locations[0] = "";

//This is your starting index. First we start off at Home!
var index = 0;

//This function will help us toggle through our photos. Everytime the button is pressed it will call this function!
function showImage() {
}
```

- `images` ist ein Array, das Bild-Adressen oder Bilder-Links für alle Fotos enthält, die du hinzufügen möchtest.
- `locations` ist ein Array, das die geografischen Standorte von jedem Bild enthält (oder die Beschreibung, falls du ein anderes Thema verwenden möchtest).

Da wir jedes Foto und seinen Titel zu den Arrays `images` und `locations` hinzufügen, wollen wir sicherstellen, dass die Indexwerte in beiden Arrays gleich sind. Das bedeutet, wenn ich ein Bild von den Nordlichtern in das Array `images` bei Index 0 stelle, werde ich auch den Titeltext „Nordlichter“ im Array `locations` bei Index 0 hinzufügen.

Um ein Foto zum Array hinzuzufügen, nehmen wir anstatt dessen die **Bild-URL-Adresse**.

- **Wenn du Bilder online verwendest, solltest du die Bild-URL-Adresse in deinem Planungsleitfaden speichern.** Wie du eine Bild-Adresse erhältst, erfährst du in diesem [Video](#).
- **Wenn du deine eigenen Bilder verwendest, achte darauf, dass du deine Bilder *digital* auf deinem Computer gespeichert hast.** Schaue dir zum Hochladen von Bildern in Glitch dieses [Video](#) an.

Da du nun die URL-Adressen deiner Bilder hast, füge sie zu deinem Array `images` hinzu. Navigiere dann zurück zur Datei `script.js`. Da dies dein erstes Bild ist, fügen wir die Bild-URL bei Index 0 hinzu. Füge deine Bild-URL-Adresse zwischen `" "` auf Zeile 6 in deinem Startercode hinzu.

```
1~ /* If you're feeling fancy you can add interactivity
2   to your site with Javascript */
3
4 //This is an array that will hold the file names! Add at least 4 images to the images array.
5 var images = new Array();
6 images[0] = "https://cdn.glitch.com/5b718117-0dff-4b23-9a99-5b6ff3f4ee3d%2FNorthern%20Lights.jpg?v=1586357262455";
7 //This is an array that holds the names of each location shown in your pictures.
8 //Make sure that the index of each location is the same as the index of the picture in the images array
9 var locations = new Array();
10 locations[0] = "";
11
12 //This is your starting index. First we start off at Home!
13 var index = 0;
14
15 //This function will help us toggle through our photos. Everytime the button is pressed it will call this function!
16~ function showImage() {
17
18 }
19
```

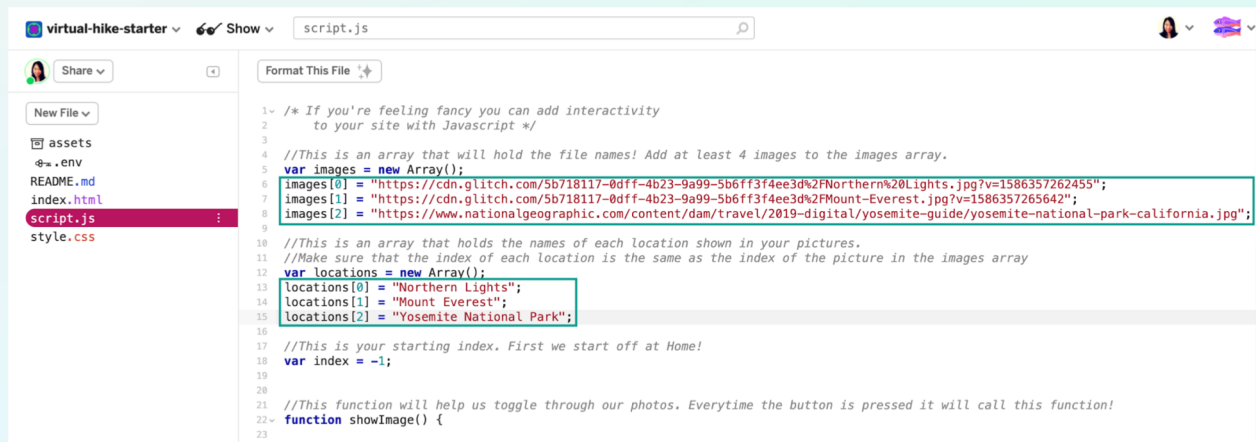
## Schritt 9: Füge dein erstes Bild und den ersten Standort hinzu (Fortsetzung)

Als nächstes möchten wir den Standort oder den Titel des gerade hochgeladenen Bildes hinzufügen. Füge den Titel des Bildes hinzu, das du gerade zwischen die doppelten Anführungszeichen " " zum Index 0 des Arrays `locations` hochgeladen hast.

```
1~ /* If you're feeling fancy you can add interactivity
2   to your site with Javascript */
3
4 //This is an array that will hold the file names! Add at least 4 images to the images array.
5 var images = new Array();
6 images[0] = "https://cdn.glitch.com/5b718117-0dff-4b23-9a99-5b6ff3f4ee3d%2FNorthern%20Lights.jpg?v=1586357262455";
7 //This is an array that holds the names of each location shown in your pictures.
8 //Make sure that the index of each location is the same as the index of the picture in the images array
9 var locations = new Array();
10 locations[0] = "Northern Lights";
11
12 //This is your starting index. First we start off at Home!
13 var index = 0;
14
15 //This function will help us toggle through our photos. Everytime the button is pressed it will call this function!
16~ function showImage() {
17
18 }
19
```

## Schritt 10: Füge mehr Bilder und Standorte hinzu (10 Min.)

Da du deine ersten Bilder und Standorttitel hinzugefügt hast, solltest du nun deine verbleibenden Bilder hochladen. Folge den Anweisungen in Schritt 9, um mehr Bilder und Text hinzuzufügen. Bedenke beim Hinzufügen der Bild-URL-Adresse zum Array `images`, dass du den Bildtitel in das Array `locations` **im gleichen Index hinzufügen musst**. Wenn all deine Bilder und Standorte hinzugefügt wurden, sollte dein Code so ähnlich wie auf dem folgenden Bild aussehen.



```
1~ /* If you're feeling fancy you can add interactivity
2   to your site with Javascript */
3
4 //This is an array that will hold the file names! Add at least 4 images to the images array.
5 var images = new Array();
6 images[0] = "https://cdn.glitch.com/5b718117-0dff-4b23-9a99-5b6ff3f4ee3d%2FNorthern%20Lights.jpg?v=1586357262455";
7 images[1] = "https://cdn.glitch.com/5b718117-0dff-4b23-9a99-5b6ff3f4ee3d%2FMount-Everest.jpg?v=1586357265642";
8 images[2] = "https://www.nationalgeographic.com/content/dam/travel/2019-digital/yosemite-guide/yosemite-national-park-california.jpg";
9
10 //This is an array that holds the names of each location shown in your pictures.
11 //Make sure that the index of each location is the same as the index of the picture in the images array
12 var locations = new Array();
13 locations[0] = "Northern Lights";
14 locations[1] = "Mount Everest";
15 locations[2] = "Yosemite National Park";
16
17 //This is your starting index. First we start off at Home!
18 var index = -1;
19
20
21 //This function will help us toggle through our photos. Everytime the button is pressed it will call this function!
22~ function showImage() {
23
24 }
```

Du hast nun die Bilder hinzugefügt, wenn du jedoch auf die Schaltfläche „Weiter“ klickst, geschieht nichts. Es ist uns nun gelungen, diese Bilder zu speichern, aber wir haben noch keinen Befehl eingerichtet, um darauf zuzugreifen. Lass uns das ändern.

## Schritt 11: Einführung in Funktionen (5 Min.)

Schauen wir uns erstmal das Tag „button“ an. Navigiere zur Datei `index.html`, dort findest du diese Codezeile.

```
<button type="button" onclick="showImage()">Next</button>
```

Beachte, dass das Attribut `onclick showImage()` entspricht. Das Attribut `onclick` wird verwendet, um dem Computer mitzuteilen, was er zu tun hat, wenn die Schaltfläche geklickt wird. In diesem Fall ruft unsere Schaltfläche die **Funktion** `showImage()` auf oder führt sie aus. Die Funktion `showImage()` ist in unserer JavaScript-Datei enthalten. Bevor wir uns damit befassen, was `showImage()` ausführt, besprechen wir, was eine Funktion ist.

Eine **Funktion** ist ein benannter Bereich eines Codes, der eine Reihe von Anweisungen ausführt (oder Codezeilen). Wenn z. B. jemand zu dir sagt „Decke den Tisch“, weißt du gleich, was das bedeutet: ein Gedeck für die Anzahl der Personen bei dir zu Hause auf dem Tisch vorzubereiten, dann an jedem Platz die Utensilien anzurichten (Gabel, Löffel, Messer und/oder Essstäbchen) und eine Serviette danebenzulegen. **Decke den Tisch** ist eine **Funktion** und die einzelnen, für das Tischdecken erforderlichen Aufgaben sind in dieser Funktion enthalten. Natürlich können die Anweisungen zum Tischdecken bei dir zu Hause anders sein. Als Programmierer\*innen können wir die Aufgaben, die in einer Funktion beschrieben sind, *näher benennen*.

Schauen wir uns die Funktion `showImage()` in der Datei `script.js` an.

```
1- /* If you're feeling fancy you can add interactivity
2   to your site with Javascript */
3
4 //This is an array that will hold the file names! Add at least 4 images to the images array.
5 var images = new Array();
6 images[0] = "https://cdn.glitch.com/5b718117-0dff-4b23-9a99-5b6ff3f4ee3d%2FNorthern%20Lights.jpg?v=1586357262455";
7 images[1] = "https://cdn.glitch.com/5b718117-0dff-4b23-9a99-5b6ff3f4ee3d%2FMount-Everest.jpg?v=1586357265642";
8 images[2] = "https://www.nationalgeographic.com/content/dam/travel/2019-digital/yosemite-guide/yosemite-national-park-california.jpg";
9
10 //This is an array that holds the names of each location shown in your pictures.
11 //Make sure that the index of each location is the same as the index of the picture in the images array
12 var locations = new Array();
13 locations[0] = "Northern Lights";
14 locations[1] = "Mount Everest";
15 locations[2] = "Yosemite National Park";
16
17 //This is your starting index. First we start off at Home!
18 var index = -1;
19
20
21 //This function will help us toggle through our photos. Everytime the button is pressed it will call this function!
22 function showImage() {
23
24 }
```

Die Verwendung der **Funktion keyword** informiert das Programm, dass du eine Funktion deklarierst. Auf das Schlüsselwort folgen der Name der Funktion und alle **Parameter** – oder Informationseinheiten, die an die Funktion als Input weitergegeben werden müssen – und die zwischen der Klammer `()` stehen. Vielleicht hast du festgestellt, dass in unserer Funktion nichts zwischen den beiden Klammern steht. Das ist deshalb so, weil Parameter optional sind. Alle beim **Aufrufen** der Funktion erforderlichen Schritte sind in einem **Codeblock** zwischen den Klammern `{ }` enthalten, der mit dem Satzzeichen `;` abschließt.

Wir möchten nun, dass die Funktion Folgendes ausführt:

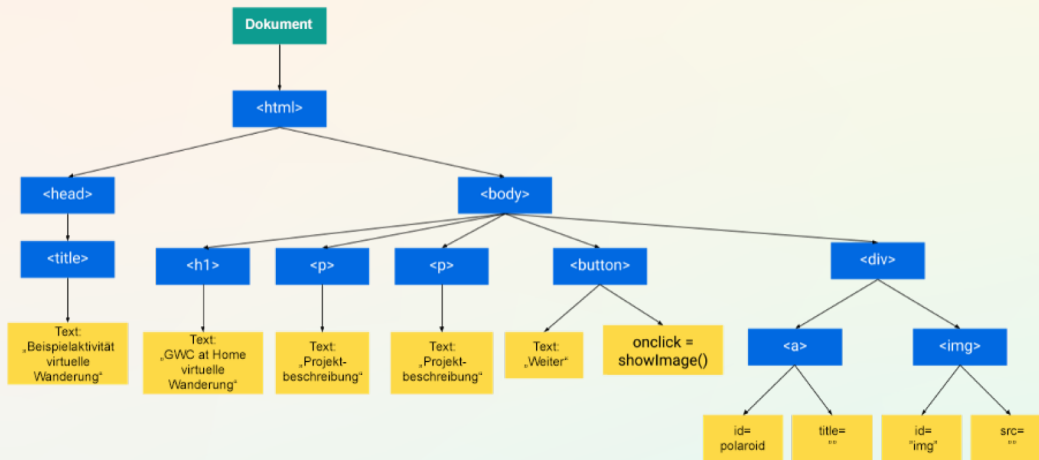
1. Das Bild im Polaroid zu aktualisieren
2. Den Text im Polaroid zu aktualisieren
3. Zum nächsten Bild weiterzugehen

In den nächsten drei Schritten schreiben wir den Code, um unsere virtuelle Wanderung interaktiv zu machen.

## Schritt 12: Aktualisiere das Bild im Polaroid (10 Min.)

Um das Bild im Polaroid zu aktualisieren, müssen wir das DOM verwenden. Eine Website verwendet ein **DOM** (Dokumentobjektmodell), um den gesamten Inhalt einer Website zu organisieren. Das DOM steht für Struktur, die in unserer HTML-Datei enthalten ist. Hier ist eine DOM für unsere derzeitige Website.

```
<div class="polaroid">
  <a id="polaroid" href="#" title="">
    <img id="img" src="" />
  </a>
</div>
```



Vielleicht merkst du, dass die **blauen** Kästchen für die Tags in der HTML-Datei stehen. Die in den **gelben** Kästchen dargestellten Informationen sind **Attribute** für jeden Elementtyp. Attribute sind Informationen, die für jedes Element eindeutig sind. Genau wie deine Haarfarbe, deine Augenfarbe und dein Name Attribute von dir sind. Damit unsere JavaScript-Datei mit unserem HTML-Gerüst kommuniziert, verwenden wir DOM, um die Elemente in den Dateien `script.js` und `index.html` zu verknüpfen. Man könnte sagen, dass das DOM für unseren JavaScript-Code eine Zuordnung ist, um die Elemente zu suchen, die wir verändern möchten. Die Verwendung der **ID** eines Elements ist die einfachste Art, ein Element in deiner HTML-Datei aufzurufen. Beachte, dass das Element, das auf das Polaroidfoto verweist, den ID-Namen **polaroid** hat und dass das Bild in dem Polaroidfoto über ein Tag `<img>` und einen ID-Namen **img** verfügt.

**Hinweis:** Die nächsten Codezeilen sollten *in* die Funktion `showImage()` geschrieben werden. Der gesamte Code sollte zwischen die Klammern `{ }` geschrieben werden.

### 1. Verwende das DOM, um einen Verweis auf das Bild-Tag mit der ID „img“ zu erstellen.

Wir müssen unserer JavaScript-Datei mitteilen, welches Element wir in der HTML-Datei ändern möchten. An dieser Stelle ist das DOM wirklich nützlich.

```
var img = document.getElementById("img");
```

Wir verwenden das Objekt `document`, um JavaScript mitzuteilen, dass wir auf unserer Website eine Aktion durchführen möchten. Die Methode `getElementById()` dient dazu, auf unserer Website ein Element anhand seiner Tag-ID zu finden. Wir verwenden die ID des Bildtags: „img“ um auf das Tag `<img>` zu verweisen. Schließlich speichern wir den Verweis zum Bildtag in einer Variablen, die wir `img` genannt haben.

## Schritt 12: Aktualisiere das Bild im Polaroid (Fortsetzung)

2. Aktualisiere die Bildquelle auf eines der Fotos im Array `images`. Wir haben bereits eine Variable erstellt, die Index heißt und bei 0 beginnt. So wird das Bild, das du zeigen willst, nachverfolgt.

```
img.src = images[index];
```

Wir haben unser Objekt `img`, das wir über das DOM gefunden haben. Nun möchten wir ein Quellenattribut hinzufügen, indem wir das Schlüsselwort `src` verwenden. Hier nutzen wir `images[index]`, um das gewünschte Foto zu finden, das wir mit dem in der Variable `Index` gespeicherten Indexwert verwenden möchten. Wir verwenden das Gleichheitszeichen `=`, um die Bildquelle dem neuen Bild zuzuweisen.

```
//This function will help us toggle through our p  
function showImage() {  
  var img = document.getElementById("img");  
  img.src = images[index];  
}
```

3. Prüfe, ob dein Code funktioniert! Klicke in unserer Live-Website auf die Schaltfläche „Weiter“. Dein erstes Bild sollte nun angezeigt werden. Falls nicht, prüfe noch einmal, ob am Ende einer jeden Codezeile in der Funktion Semikolons stehen und ob deine Bilder die richtige URL-Adresse haben.

## Schritt 13: Aktualisiere den Text im Polaroid (5 Min.)

Nun fahren wir fort und programmieren in der Funktion `showImage()`, indem wir einen Code hinzufügen, um den mit dem Bild gezeigten Text zu aktualisieren. Beachte, dass all diese Schritte zwischen die Klammern `{ }` geschrieben werden müssen.

1. Erstelle mit dem DOM einen Verweis zum Polaroid-Tag, indem du die Id „`polaroid`“ verwendest.

```
var polaroid = document.getElementById("polaroid");
```

Hier rufen wir das DOM mit dem Schlüsselwort `document` auf und verweisen auf das Bild-Tag mit seiner ID `"polaroid"`. Da wir die ID verwenden, um das Tag abzurufen, verwenden wir die Methode `getElementById()`. Schließlich speichern wir das in einer Variablen, die wir `polaroid` genannt haben.

2. Aktualisiere den Titel von einem der Standorte im Array `locations`. Wir haben bereits eine Variable erstellt, die Index heißt und mit 0 beginnt. Das geschieht, damit nachverfolgen kannst, welche Beschreibung du zeigen möchtest.

```
polaroid.title = locations[index];
```

Hier nutzen wir `locations[index]`, um das Foto abzurufen, das wir mit dem in der Variable `Index` gespeicherten Indexwert verwenden möchten. Wir müssen sicherstellen, den gleichen Index zu verwenden, um für jedes Bild den richtigen Standort abzurufen. Als Nächstes rufen wir das Titel-Attribut ab, indem wir die Eigenschaft `title` verwenden. Wir verwenden das Gleichheitszeichen `=`, um den Text dem richtigen Standort zuzuweisen.

```
function showImage() {  
  var img = document.getElementById("img");  
  img.src = images[index];  
  
  var polaroid = document.getElementById("polaroid");  
  polaroid.title = locations[index];  
}
```

## Schritt 14: Gehe weiter zum nächsten Bild (2 Min.)

Jedes Mal, wenn du die Schaltfläche „Weiter“ klickst, sollte ein Bild angezeigt werden. Aber im Moment wird nur eines der vier Bilder angezeigt. Warum? Um das gewünschte Bild und die in unseren Arrays gespeicherten Daten abzurufen, haben wir die Variable **Index** verwendet. Bisher ist der **Index** auf 0 gesetzt, aber wir haben das nicht geändert.

Nachdem wir das Bild und den Titel aktualisiert haben, möchten wir den Indexwert um 1 erhöhen, sodass wir jedes Foto und jeden Standort durchgehen können.

```
index = index + 1;
```

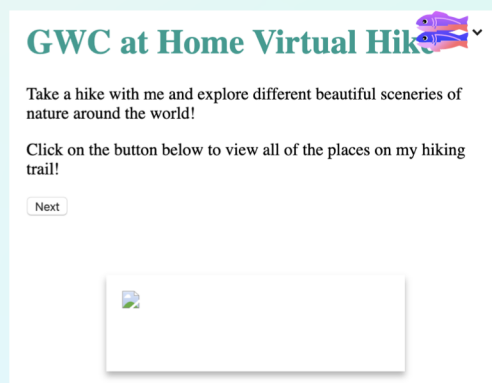
Hier weisen wir den Indexwert erneut zu, sodass er genau dem vorherigen Wert plus 1 entspricht. Wir wollen dies als die **letzte** Anweisung in dieser Funktion hinzufügen.

```
function showImage() {  
  var img = document.getElementById("img");  
  img.src = images[index];  
  
  var polaroid = document.getElementById("polaroid");  
  polaroid.title = locations[index];  
  index = index+1;  
}
```

Lass uns schnell testen, ob unser Code funktioniert! Klicke in unserer Live-Website jetzt auf die Schaltfläche „Weiter“. Jedes Mal, wenn du die Schaltfläche „Weiter“ klickst, solltest du in deiner Bildschirmpräsentation das nächste Bild sehen.

## Schritt 15: Hoppla! Zu weit gegangen! (5 Min.)

Vielleicht hast du festgestellt, dass eine Fehlermeldung angezeigt wird, wenn du nach deinem letzten Bild wiederholt auf die Schaltfläche „Weiter“ klickst.



Da wir unseren **Indexwert** bei jedem Klicken auf die Schaltfläche „Weiter“ erhöhen, versucht unser Programm schließlich Daten aus unserem Array zu holen, die nicht existieren. Deshalb müssen wir irgendwie herausfinden, ob unser Index zu lang ist. Eine **Bedingungsanweisung** prüft, ob eine Gruppe von Regeln (oder eine Anweisung) erfüllt wird und entscheidet dann, welche Aktionen ausgeführt werden, falls die Regelgruppe oder die Anweisung wahr oder falsch ist. In unserem Fall möchten wir wissen, ob der **Index** zu groß ist – mit anderen Worten, ob er die Länge/Größe des Arrays überschreitet.

## Schritt 15: Hoppla! Zu weit gegangen! (Fortsetzung)

Bevor wir unsere Bedingungsanweisung schreiben, schauen wir uns die möglichen Fälle an.

1. Falls der **Index** nicht zu groß ist (kürzer als die Länge des Arrays):  
Dann können wir unseren Index wie zuvor um 1 erhöhen.
2. Falls der **Index** zu groß ist:  
Dann sollten wir den **Index** so lassen, wie er ist. Wenn wir den **Indexwert** nicht ändern, bleibt für unsere Benutzer einfach das letzte Bild sichtbar und es folgen keine weiteren Bilder.

Aber wie können wir wissen, dass der **Index** zu groß ist? Wir können das anhand der Größe des Arrays sagen. Erinnerst du dich, was der letzte Indexwert im Array ist? Welcher Zusammenhang besteht zur Größe? In der Tat hat der letzte Index des Arrays immer einen Wert *weniger* als die Größe/Länge des Arrays. Das ist so, weil Arrays einen Index von 0 haben. Wir wissen nun, dass der letzte Index durch `images.length - 1` dargestellt werden kann.

Beim Schreiben von Bedingungen verwenden wir die Schlüsselwörter **if** und **else**, um unser Regelset aufzuschlüsseln und verwenden den Codeblock `{ }`, um dem Computer mitzuteilen, welche Codezeile er ausführen soll, wenn die Bedingung wahr ist.

Wir möchten unsere vorherige Anweisung `index = index + 1`; durch die folgenden Codezeilen ersetzen, um herauszufinden, ob der Index zu groß sein könnte.

```
function showImage() {  
  var img = document.getElementById("img");  
  img.src = images[index];  
  
  var polaroid = document.getElementById("polaroid");  
  polaroid.title = locations[index];  
  
  if (index < images.length - 1) {  
    index = index + 1;  
  }  
}
```

Um zu sehen, ob der **Index** zu groß ist, vergleichen wir ihn mit der **Länge** des Arrays. Da wir gesagt haben, dass der letzte Index in dem Array immer einen Wert weniger hat als die Länge, haben wir diesen Wert in unserer Bedingung (oder Regel) verwendet. Wir wissen, wenn der Index nicht zu groß ist, erhöhen wir den **Indexwert** um 1. Wir fügen keinen Code mehr hinzu, um den zweiten Fall zu bestimmen, in dem der **Index** zu groß ist. Denn in diesem Fall wollen wir den **Indexwert** nicht aktualisieren. Mit anderen Worten, wenn der **Index** zu groß ist, tun wir nichts.

Wir nehmen uns nun ein paar Minuten Zeit, um deinen Code zu testen. Achte darauf, die Schaltfläche „Weiter“ mehrere Male zu **klicken**, um zu testen, ob deine Schaltfläche beim letzten Bild nicht mehr reagiert.

## Schritt 16: Erweiterungen (5–20 Min.)

### Füge ausgefallene Schriften hinzu (5–10 Min.)

In diesem Projekt legen wir die Schrift „Roboto“ fest, aber du kannst dies in deinem Projekt ändern. Du kannst ausgefallene Schriften ganz leicht aus [Google Fonts](#) importieren und in deiner Website verwenden. Schau dir dieses [Video](#) an, um zu erfahren, wie du Schriften in deiner Website ändern kannst.

### Ändere die Hintergrundfarbe deiner Website (5–10 Min.)

Lass uns die Hintergrundfarben durch ausgefallene Farben ändern! Farben werden numerische Werte zugeordnet. Denn so ist es für den Computer leichter, die verschiedenen Farben zu unterscheiden. Sie können durch ihren HEX-, RGB- oder HSL-Wert dargestellt werden. Wir empfehlen bei einem Typen zu bleiben, damit alle Farben konsistent sind. W3Schools bietet eine tolle [Farbauswahl](#) (Color Picker) an, mit der du die Werte der Farben für deine Website leichter finden kannst. Möchtest du zu deinem Projekt einen coolen Farbverlauf hinzufügen? Verwende diese [Website](#) als Hilfe, um den CSS-Code besser zu planen und zu erstellen. Schau dir dieses [Video](#) für detaillierte Anweisungen zur Änderung von Farben an.

### Füge mehrere CSS-Stylesheets hinzu (10–20 Min.)

Mit CSS kannst du so vieles tun, damit deine Website einzigartig wird. Wir haben eine Liste mit Ressourcen zusammengestellt, in denen du weitere Tipps für die Gestaltung deiner Website mit CSS findest.

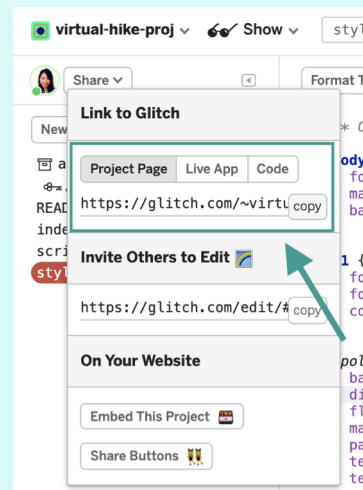
- W3Schools hat für CSS einen tollen [Referenzbereich](#).
- Mozilla bietet in seinem CSS-Modul eine Fülle von hilfreichen Informationen zu:
  - [Formatieren von Text](#)
  - [CSS-Layout](#)
  - [CSS zum Lösen von Problemen](#)
  - [Dein CSS debuggen](#)

## Schritt 18: Teilt euer Projekt von Girls Who Code at Home! (5 Min.)

Wir freuen uns auf deine Projekte mit virtuellen Wanderungen. Vergiss nicht Folgendes zu taggen [@girlswhocode](#) [#codefromhome](#). Vielleicht wirst du sogar in unserem Account vorgestellt!

### So teilst du ein Glitch-Projekt:

1. Klicke auf die Schaltfläche **Share** (Teilen) oben links neben dem Profil-Symbol.
2. Du kannst auch den Link deiner Projektseite weitergeben (damit können andere deinen Code sehen) oder einfach nur die Live-App. Klicke auf die Option, die du zum Weitergeben verwenden möchtest und kopiere den bereitgestellten Link.



# Arbeitsblatt Projektplanung Virtuelle Wanderung

## Planungsübersicht für das Projekt

**Thema:** Welches Thema soll deine virtuelle Wanderung haben?

**Publikum:** Für wen ist diese Website gedacht? Welche Personengruppe wird an deinem Produkt Interesse haben?

**Ziel:** Was möchtest du mit deiner Website erreichen? Warum ist das für dein Publikum von Interesse?

## Bildplanung

Wähle mindestens drei Bilder aus, die in deiner virtuellen Wanderung enthalten sein sollen. Mache dir über die **Reihenfolge** deiner Bilder Gedanken. Fülle die untenstehende Tabelle aus und gib das Bild, die Bild-URL-Adresse und den Titel an.

Nummer	Bild	Bildquelle: <i>(Gib an, ob es dein eigenes Foto oder ein Foto von einer anderen Website ist. Füge bei einem Foto von einer anderen Website einen Link zur Seite bei.)</i>	Titel <i>(Welchen Text möchtest du deinem Publikum zu diesem Bild zeigen?)</i>
0.			
1.			
2.			
3.			

## Bildplanung (Fortsetzung)

Nummer	Bild	Bildquelle: (Gib an, ob es dein eigenes Foto oder ein Foto von einer anderen Website ist. Füge bei einem Foto von einer anderen Website einen Link zur Seite bei.)	Titel (Welchen Text möchtest du deinem Publikum zu diesem Bild zeigen?)
4.			
5.			
6.			
7.			