NSC-Generator

für Pen & Paper-Systeme

[Kurzbeschreibung](#_8faq8utfwria)

[Warum ein NSC-Generator?](#_31f24kc98esr)

[Was macht die Anwendung?](#_4h5gawnu9r5a)

[Aktuelle Notizen](#_2pmp857fwknc)

[Funktionen](#_9r8ucoy5zb9y)

[HTML-Ausgabe](#_8t581wh47kuy)

[Variablen](#_5rn30emvitp9)

[Funktionen](#_hry4pkp0o56p)

[Javascript](#_y3bn5ehfpg35)

[Variablen](#_vie4y1plw1fx)

[Funktionen](#_84z0k75twkhf)

[Programmablauf](#_65u8ej8duvx3)

[Von der Seite des Programmierers...](#_jzc3s7lpwq7)

[Was ist ein Fragenpool?](#_nst5972oavb5)

[Festlegung des IGFP (individueller Grundfragenpool)](#_u38np7iz9n5l)

[Was passiert mit dem Grundfragenpool im Laufe der Generierung?](#_objgwbn5w3oy)

[Werteberechnung](#_ki6eyz3jymkz)

[Weitere mögliche Funktionen für später, wenn die Grundoptionen funktionieren](#_ptr9ect15v6h)

[Von der Seite des Nutzers...](#_mfzk1gk9siph)

[Für die Oberfläche](#_htnak4ua0q16)

[Während der Charaktererstellung](#_j43ft0oq1pru)

[Beispiel](#_ej1bj485a18)

# 

# 

# 

# Kurzbeschreibung

### Warum ein NSC-Generator?

Ich bin ein großer P&P-Rollenspielfan. Als Spielleiter ist es für fast jedes eigene Abenteuer nötig, sich Nichtspielercharaktere auszudenken. Diese spielen häufig nur in einem Abenteuer mit, aber die Erstellung ist stets mit einem hohen Zeitaufwand verbunden.

### Was macht die Anwendung?

Die Anwendung richtet sich an Spielleiter von Rollenspielsystemen. Sie soll dem Nutzer bei der Erstellung der erzählerischen Anteile helfen und den kreativen Prozess, durch zur Figur passende Vorschläge und Fragestellungen, vereinfachen.

* die Anwendung geht auf Nutzereingaben ein- und gibt dem Nutzer individualisierte Ausgaben
* die Werteberechnung der Attribute erfolgt am Ende dynamisch anhand der Antworten über Würfelmodifikatoren (siehe *Werteberechnung*)

# 

# 

# Aktuelle Notizen

Tagsystembeispiel ([Link](https://www.w3schools.com/code/tryit.asp?filename=FCSZWWFC92V4))

# 

# 

# Funktionen

## HTML-Ausgabe

### Variablen

var preRole, preTypus, preGender, preSystem;

var characterName;

var preSelectionTags;

### Funktionen

**out()**

* *out(“Lorem ipsum...”);*

**change(button\_ID, newTxt)**

* *change("button2", "Hier ist ein Knopf.");*

**changeLabel(button\_ID, newTxt)**

* *changeLabel("answer01", "Hier ist ne Frage für dich!");*

**confirmPreselection()**

* Funktion, die die Vorauswahl-variablen zuordnet und sie den aktiven Tags hinzufügt

**confirmAnswer()**

* *lorem ipsum*

# 

# 

## Javascript

### Variablen

**Speicher-Array**

* var allQuestions = [];

**Aktive Tags**

* var activeTags = " ";

**Fragenzähler**

* var questionCount = 0;

**Kategorien**

* var questCat1 = [];
* ...
* var questCat8 = [];

**Zähler für die Kategorienabfrage**

* var currentCategory = 1;
* var answeredQuestionCounter = 1;

**Konstante Variablen**

* const CATEGORY\_MAX\_COUNT = 8;
* const QUESTION\_LIMIT = 7;

**Alle Fragen durch?**

* var allDone = false;

### Funktionen

**modifyTags(tagsToBeAdded)**

* Funktion, die activeTags weitere Tags hinzufügt
* *modifyTags(“#er#zwo#uno#mah”);*

**addQuestion(category, txt, exclusionTags)**

* Funktion, die allQuestions um eine Frage erweitert
* *new addQuestion(1, "Wie sind die Eltern deiner Figur ums Leben gekommen?", "parentsAlive#statist#parentsUnknown");*

**addQuestion.prototype.addAnswer(answerTxt, answerExclusionTags, tagsToAdd)**

* Funktion innerhalb von addQuestion, die es erlaubt neue Fragen in das entsprechende answerArray des addQuestions-Objekts zu speichern
* *allQuestions[1].addAnswer("Dorf", "#", "#OriginVillage");*

**checkForMatch(exclusionTags, activeTags)**

* Funktion, die überprüft, ob es eine Übereinstimmung zwischen den ausschließenden- und den aktiven Tags gibt; wenn ja, wird true ausgegeben

**updateValidity(arrayToLookThrough, activeTagsToCheck)**

* Funktion, die mithilfe von checkForMatch die Gültigkeit ändert
* *out(updateValidity(allQuestions, activeTags));*

**checkValidity(arrayToLookThrough)**

* Funktion, die nur die derzeit gültigen Fragen ausspuckt
* *out(checkValidity(allQuestions));*

**checkValidityAndCategory(arrayToLookThrough, currentCategory)**

* *Funktion, die nur die derzeit gültigen Fragen ausspuckt, zusätzlich gefiltert nach Kategorie*
* *nur für allQuestions relevant*
* *checkValidityAndCat(allQuestions, 2);*

**checkIfCatGoesUp()**

* Funktion, die überprüft ob ein Wechsel in die nächste Kategorie erfolgt ist oder ob die letzte Kategorie abgeschlossen wurde

**newCharacter()**

* Funktion, die zum Starten der Generierung ausgeführt wird
* siehe 1)
* *loremipsum*

**nextQuestion()**

* loremipsum
* *loremipsum*

**setAnswer()**

* loremipsum
* *loremipsum*

**loremipsum()**

* loremipsum
* *loremipsum*

# 

# Programmablauf

1. **Nutzer startet die Erstellung und beendet die Vorauswahl**
   1. ***newCharacter()*** sorgt dafür, dass der Kategorienzähler und der Fragenzähler zurückgesetzt werden
   2. Es startet die Vorauswahl:
      1. ***confirmPreselection()*** nimmt die Auswahl des Users an
      2. ***modifyTags()*** *= activeTags+=preSelection (mit Anti-Dopplungsfunktion)*
   3. das Array allQuestions bzw. die nächste Kategorie wird gemischt
2. **Die Gültigkeit wird angepasst** 
   1. ***updateValidity()***durchsucht allQuestions bzw. das Kategorienarray mit checkForMatch:
      1. wenn eine Übereinstimmung besteht, wird valid auf **false** gesetzt
      2. wenn keine Übereinstimmung besteht, passiert **nichts**
3. **Kategorie wird mit *checkIfCatGoesUp()* überprüft**
   1. Grundwert für cat = 1;
      1. *wenn x Fragen aus Kategorie 1 ausgegeben wurden (Counter questionCount++) ODER wenn es keine Fragen aus Kategorie 1 mehr gibt, schaltet cat++; (nochmal neu schreiben)*
4. **Ausgabe der ersten/nächsten Frage und der Antwortmöglichkeiten mit *nextQuestion()*:**
   1. Frage wird ausgegeben
   2. Antworten, die im Fragen-Objekt gespeichert sind, werden ausgegeben
      1. Ermittlung der Anzahl der Fragen, übrige radio-Felder müssen auf hidden geschaltet werden und nicht anklickbar gemacht werden
5. **User kann jetzt eine Antwort auswählen // ggf. Randombutton?**
   1. User wählt eine Antwort per Klick aus, ***showModifierChange()***;
      1. visuelle Ausgabe für den Fall einer Modifikator-Änderung erfolgt
   2. User bestätigt die Antwort mit einem anderen Knopf ***setAnswer();***
      1. ***modifyTags()*** wird ausgeführt, um activeTags um die Tags der Antwort zu erweitern
      2. Valid der Frage wird auf “false” gesetzt, sodass sie nicht noch einmal vorkommen kann
      3. Fragencounter wird um eins hochgezählt
6. **zurück zu Nummer 2)**

# Von der Seite des Programmierers...

### Was ist ein Fragenpool?

* Gruppierungen von Fragen und Antworten
* sie sind in Kategorien unterteilt bzw. thematisch sortiert (Kindheit, Moral, Ziele)
* stehen stets in einer Abhängigkeit zueinander

### Festlegung des IGFP (individueller Grundfragenpool)

* Nutzer füllt Pflichtfelder aus
* diese Angaben bestimmen den *Grundfragenpool* und die Anzahl der Fragen

*Beispiel: System/Setting (notwendig), Funktion und Gesinnung (notwendig), Rasse (optional), usw … (siehe Tabellen)*

### Was passiert mit dem Grundfragenpool im Laufe der Generierung?

* die Anwendung geht während der Frage/Antwort-Phase auf Nutzereingaben ein und bearbeitet kontinuierlich den IGFP; das führt zu einem zunehmend kleiner-werdenden Pool
* die Anwendung übermittelt die entsprechenden Werte und verkleinert den Fragenpool laufend

### Werteberechnung

* erfolgt nach der Generierung der erzählerischen Anteile
* bezieht sich direkt auf die Antworten der Fragen/Antwort-Phase; also muss hier mit Countern o.Ä. gearbeitet werden

*Beispiel: Figur stammt vom Land, geht einem körperlichen Beruf nach und treibt viel Sport oder Ähnliches – hier würde der Counter für das Attribut Stärke/Gewandtheit hochzählen und sich am Ende als positiver Modifikator auf den Würfelwurf auswirken.*

### Weitere mögliche Funktionen für später, wenn die Grundoptionen funktionieren

Kooperationsmodus

* für beliebig viele Nutzer
* richtet sich an Spieler und Spielleiter
* gemeinsam können
  + Kontakte von Spielerfiguren erstellt werden
  + neue Figuren angelegt werden

Account-Verwaltung und Bibliothek

* erstelle Figuren können in der Bibliothek gespeichert, geteilt und eingesehen werden
* Suchfunktion für öffentliche Inhalte
* Figuren sind bewertbar, kommentierbar und importierbar

# Von der Seite des Nutzers...

### Für die Oberfläche

* Kurzerklärung
* “Neuer Charakter”
* “Datenbank bearbeiten oder erweitern”

### Während der Charaktererstellung

* Pflichtfelder, die ausgefüllt werden müssen (siehe iGFP) + bestätigen
* \* nächste Seite
* Nutzer sieht eine Frage und mehrere Antwortmöglichkeiten; wählt eine Antwort aus
* Nutzer kann den Verlauf seiner Figur aktiv verfolgen; ggf. Darstellung des Wertesystems (+/-) und geschichtliche Zusammenfassung, die sich stetig erweitert, zum Ein/Ausklappen
* Tags werden für Nutzer während Charaktererstellung nicht angezeigt
  + vielleicht: kann direkt gezeigte Werte anpassen um zu seinem gewünschten Ergebnis zu gelangen

# Beispiel

Rolle und Typ bestimmen die Art des Grundfragenpools und den Detailgrad

* *Hauptfigur (Typ: Antagonist) erhält andere und detailliertere Fragen als eine neutral-gesinnte Nebenfigur*

Aufgrund der Vorauswahl von z.B. Rasse und Spielsystem fallen einige unpassende Fragen direkt zu Anfang weg

* *Wird NICHT Sci-Fi gewählt, fallen alle Fragen/Antworten weg, die ausschließlich in einen Sci-Fi-Hintergrund passen*

Anwendung merkt sich Entscheidungen des Nutzers und passt sich während der weiteren Generierung an diese an

* *“Wege” können sich verschließen oder sich bei entsprechend hoher Anzahl von passenden späteren Antworten (wieder) öffnen*

Anwendung ist in der Lage Vorschläge zu machen und Konflikte hervorzuheben

* *das geschieht in speziellen Fällen dann, wenn sie merkt, dass eine spätere Auswahl eine frühere revidieren könnte*