
Webengineering

By rac

Published: 14.01.2008 - 15:34



Stoff zum Fach Webengineering durchgehört durch die Herren Trolese und Schröder an der HSZ-T

Related Links

- [Wiki Seite mit mehr infos zum Fach](#)
- [Project NumGuess](#)

[Project NumGuess >](#)

Project NumGuess

By rac

Published: 14.01.2008 - 15:37



Unsere Aufgabe ist es eine Applikation in PHP/Ajax zu schreiben, die eine Zahl zufällig generiert und mit Benutzereingaben vergleicht. Durch vergleichen auf dem Server wird dem Benutzer als Antwort zur eingegebenen Zahl "too low", "too high" oder "right" zurückgegeben.

Related Links

- [NumGuess Applikation](#)
- [Implementation in Drupal](#)
- [Applikations-Webseite](#)
- [Screenshots der Applikation](#)
- [Code Repository](#)
- [Sourcen herunterladen](#)

- [Requirements](#)
- [Gruppe](#)
- [TODO's](#)
- [Dokumentation](#)

[< Webengineering](#) [up Requirements >](#)

Requirements

By rac

Published: 14.01.2008 - 15:38

- [Draft Usecases](#)
- [Anforderungen von Trolese für eine gute Note](#)
- [Präsentations- Abgabetermin](#)

[< Project NumGuess](#) [up](#) [_Draft Usecases >](#)

Draft Usecases

By rac

Published: 14.01.2008 - 15:39

- [UC01 Name](#)
- [UC02 Raten](#)

[< Requirements](#) [up](#) [UC01 Name](#) [>](#)

UC01 Name

By rac

Published: 14.01.2008 - 15:40

Description

Benutzer gibt Namen ein, der später für die Hall of Fame verwendet wird.
Das ganze wird in einer Session gespeichert die auf dem Server läuft.

Actors

- User

Auslöser

User Besucht die Webseite.

Preconditions

none

Postconditions

Session wurde generiert und der Name des users darin gespeichert.
Spiel beginnt...

Normal flow

- Benutzer tippt seinen namen ein
- Benutzer drückt <return>
- Spiel wird gestartet

Alternate flow

none

Exceptions

none

[< Draft Usecases](#) [up UC02 Raten](#) >

UC02 Raten

By rac

Published: 14.01.2008 - 15:41

Description

Benutzer kann Zahl eingeben, diese wird auf dem Server mit der Sessiongenerierten Zahl verglichen. Als Antwort erhält er entweder too low, too high oder right

Actors

- User

Auslöser

Game wird gestartet.

Preconditions

Es wurde eine Session erstellt und der User hat seinen Namen angegeben.

Postconditions

Die count variable in der session des users wurde incrementiert und auf dem Bildschirm wird das Resultat angezeigt.

Normal flow

- Benutzer gibt Zahl in Eingabefeld ein
- Benutzer drückt button senden
- User sieht das Resultat des vergleiches mit der generierten Zahl auf dem Bildschirm

Alternate flow

none

Exceptions

none

[< UC01 Name up Anforderungen von Trolese für eine gute Note >](#)

Anforderungen von Trolese für eine gute Note

By rac

Published: 14.01.2008 - 15:44

- Eigenes Sessionhandling einbauen

--> Umgesetzt durch die Session class

- Http posts verwenden

--> Umgesetzt durch Ajax calls, die PHP Seiten aufrufen und Post Variablen per URL encoding übergeben

- Hall of Fame

--> Umgesetzt durch abspeicherung bei treffer in hof.dat

[< UC02 Raten up Präsentations- Abgabetermin >](#)

Präsentations- Abgabetermin

By rac

Published: 14.01.2008 - 15:48

12.11.2007 (Planung Fertigstellung der Arbeit spätestens 09.11.2007)

[< Anforderungen von Trolese für eine gute Note](#) [up](#) [Gruppe](#) >

Gruppe

By rac

Published: 14.01.2008 - 15:50

1.

```
$Gruppe['member1']="Kaya Yuecel";
```

2.

```
$Gruppe['member2']="Antonio Mastroberti";
```

3.

```
$Gruppe['member3']="Michel Racic";
```

[< Präsentations- Abgabetermin up TODO's >](#)

TODO's

By rac

Published: 14.01.2008 - 15:52

Was?	wer?	Status
Applikationsrahmen HTML	Michel	done
Login formular	Antonio	done
Enhance Login formular	Willy	done
eigenes Session Manager	Michel	done
init - SSN erstellen	Michel	done
init - Random nummer generieren	Kaya	done
init - Name in session speichern	Antonio	done
Game formular	Kaya und Antonio	done
GF - versuche anzeigen	Kaya und Antonio	done
GF - zahlen weitergeben	Kaya und Antonio	done

GF - Resultat ausgeben Kaya und Antonio done

Hall of Fame - anzeigen Michel done

Hall of Fame - abspeichern Michel done

[< Gruppe](#) [up](#) [Dokumentation](#) >

Dokumentation

By rac

Published: 14.01.2008 - 15:54

Nur die wichtigsten Aspekte

- [Diagramme](#)
- [Core functions](#)
- [Communication Methods](#)

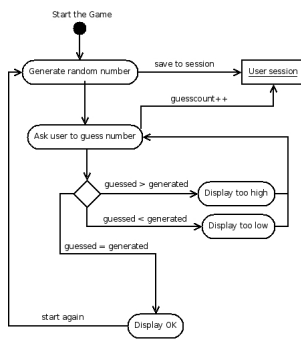
[< TODO's up Diagramme >](#)

Diagramme

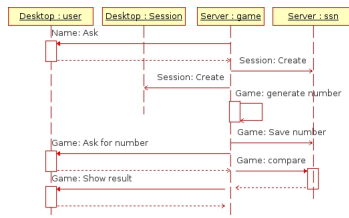
By rac

Published: 14.01.2008 - 16:04

Game-Logic



Overall Sequenz Diagramm



[< Dokumentation up Core functions >](#)

Core functions

By rac

Published: 14.01.2008 - 16:12

Session Management

Wird in der Datei [session.class.php](#) abgehandelt.

Die Dokumentation dazu befindet sich direkt im code.

Um die Session zu benutzen, muss im jeweiligen script

1.
`$Session = new Session();`

2.
`$sid = $Session->start();`

angegeben werden, danach ist der Hash des clients in der Variable `$sid` gespeichert.

Wenn ein cookie mit dem entsprechenden hash auf dem client vorhanden ist, wird diese session fortgesetzt, ansonsten wird eine neue Session initialisiert.

Um zu verhindern, dass der Session inhalt Plaintext ausgelesen werden kann, wird eine simple Methode namens Base64 encoding verwendet.

Das Array wird Serialisiert und der Serialisierte String danach mit base64 encodiert beim umgekehrten fall, das ganze vice versa.

1.
Als Tipp am rande:

2.
Damit kann man z.B. die MS Terminal Server Beschränkung umgehen,

3.
die es nicht erlaubt binäre Daten vom TS auf den client zu kopieren,

4.
da sie nach dem base64 encoding als normaler ASCII text gehandhabt werden

5.
und per <ctrl>-C / <ctrl>-V der Inhalt kopiert werden kann.

In unserer Demo applikation, sind die ssn Dateien in einem Unterordner gespeichert. Im Produktiven fall ist es sehr zu empfehlen diesen Ordner Ausserhalb des Webroots zu erstellen.

Desweiteren fehlt dieser Klasse noch das Hardening, sprich es ist noch sehr einfach ein Session Hijacking durchzuführen.

Dies haben wir aus Zeitgründen weggelassen.

[< Diagramme up Communication Methods >](#)

Communication Methods

By rac

Published: 14.01.2008 - 16:14

AJAX calls

Der JS code für die AJAX calls ist im Header der Datei [index.html](#) abgelegt.

Die Funktionen `PHPscriptCall(url, resultDiv)`, `loadInBackground(url, resultDiv)` und `getAnswer(url)` machen die jeweiligen HTTP/Post requests.

Sie sind nahezu identisch bis auf die Kleinigkeit, dass bei der Funktion `PHPscriptCall` während dem Laden eine ein Waiting Layer (DIV) auf der Seite angezeigt wird mit einem Animierten Matrix Tux darin und bei der `loadInBackground` nicht.

Die Funktion `getAnswer` macht einen call auf die jeweilige URL und wertet danach die Antwort in einer anderen Funktion aus um eine Spezifizierte Aktion auszuführen.

[< Core functions up](#)