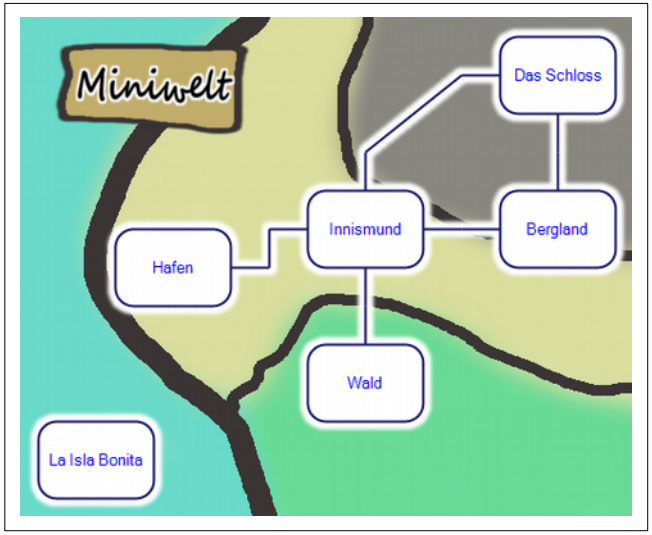
Modulares Geschichtenerzählen mit endlichen Automaten



TdI 2017 München

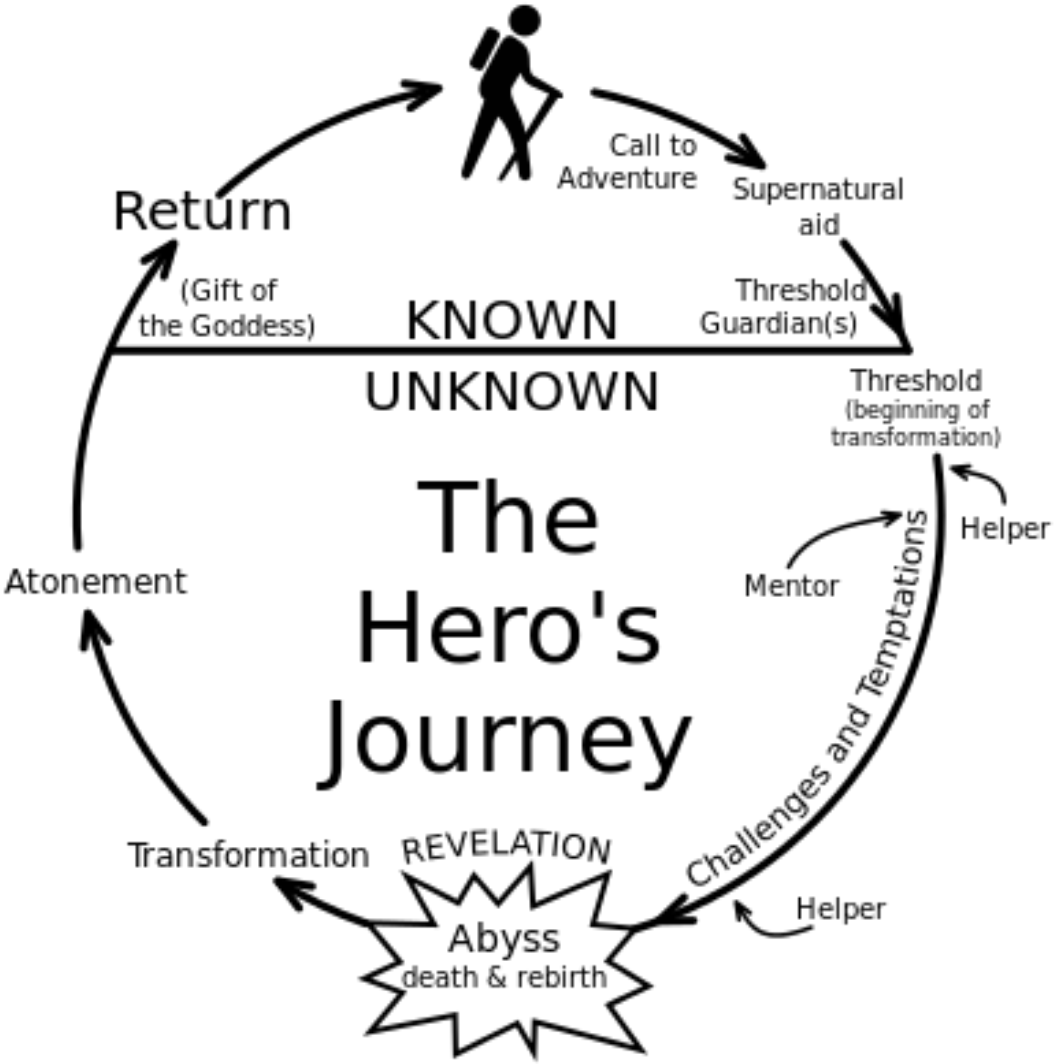
Thomas Rau
LMU München
Graf-Rasso-Gymnasium Fürstenfeldbruck
thomas.rau@ifi.lmu.de

Überblick

- Auslöser 1: Literaturtheorie
- Auslöser 2: Ein Computerspiel
- Ergebnis Projekt Klasse 10
- Eigene Versuche
- Einsatzmöglichkeiten

Ein wenig Literaturtheorie

Die Heldenreise (Joseph Campbell)



https://en.wikipedia.org/wiki/Hero's_journey#/media/File:Heroesjourney.svg – public domain

Heldenreise in Star Wars

Gewohnte Welt

Ruf zum Abenteuer

Verweigerung

Begegnung mit Mentor

Überschreiten der Schwelle

Prüfungen, Helfer, Feinde

Vordringen in tiefste Höhle

Zentrale Prüfung Transformation

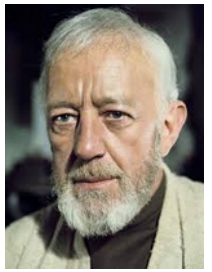
Belohnung

Rückweg

Rückkehr mit Geschenk

Gewohnte Welt

Andere Welt



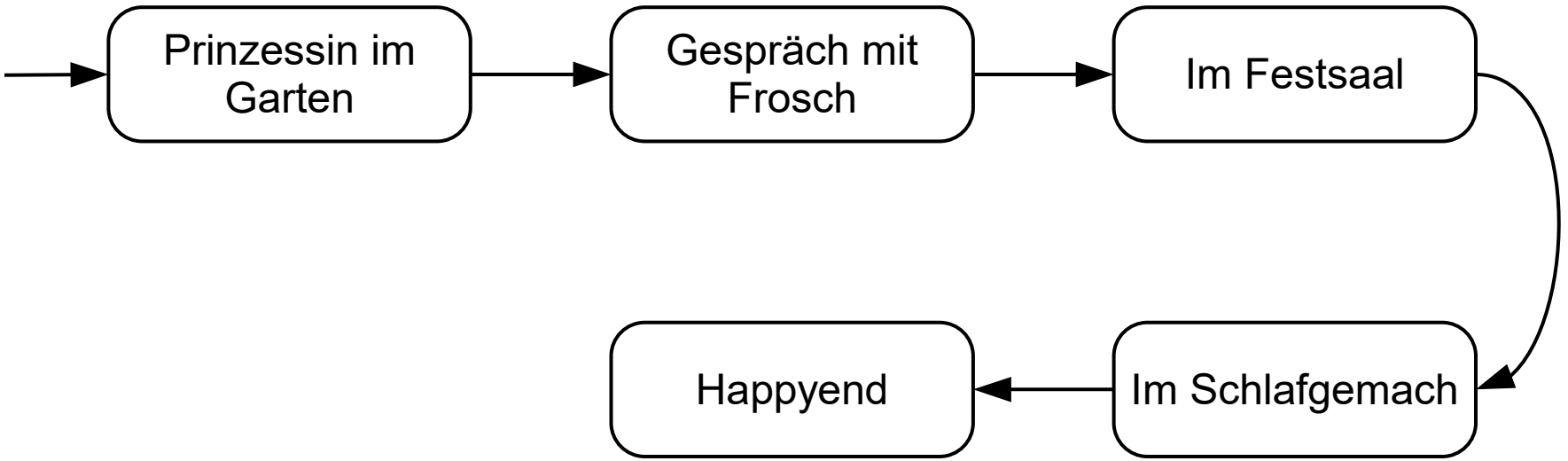
Heldenreise im Hobbit

Alter Hut für Deutschlehrer

- “Teile die Geschichte in Abschnitte ein.”

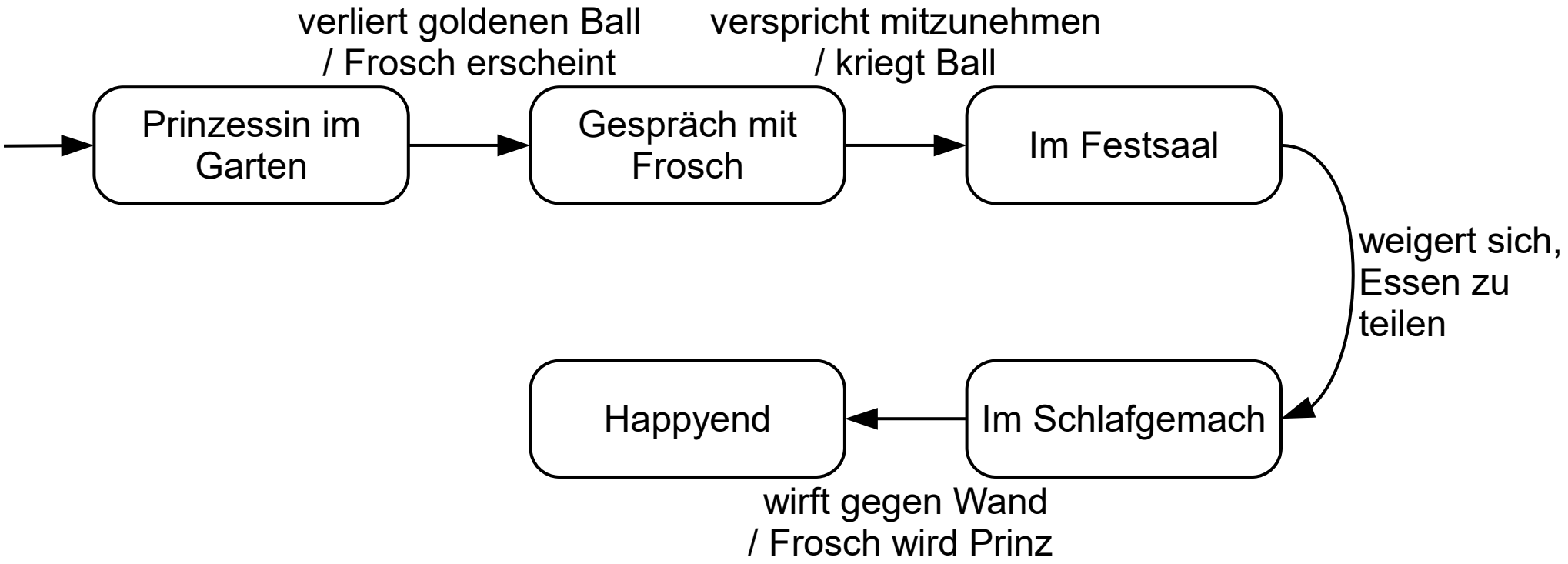
Alter Hut für Deutschlehrer

- “Teile die Geschichte in Abschnitte ein.”



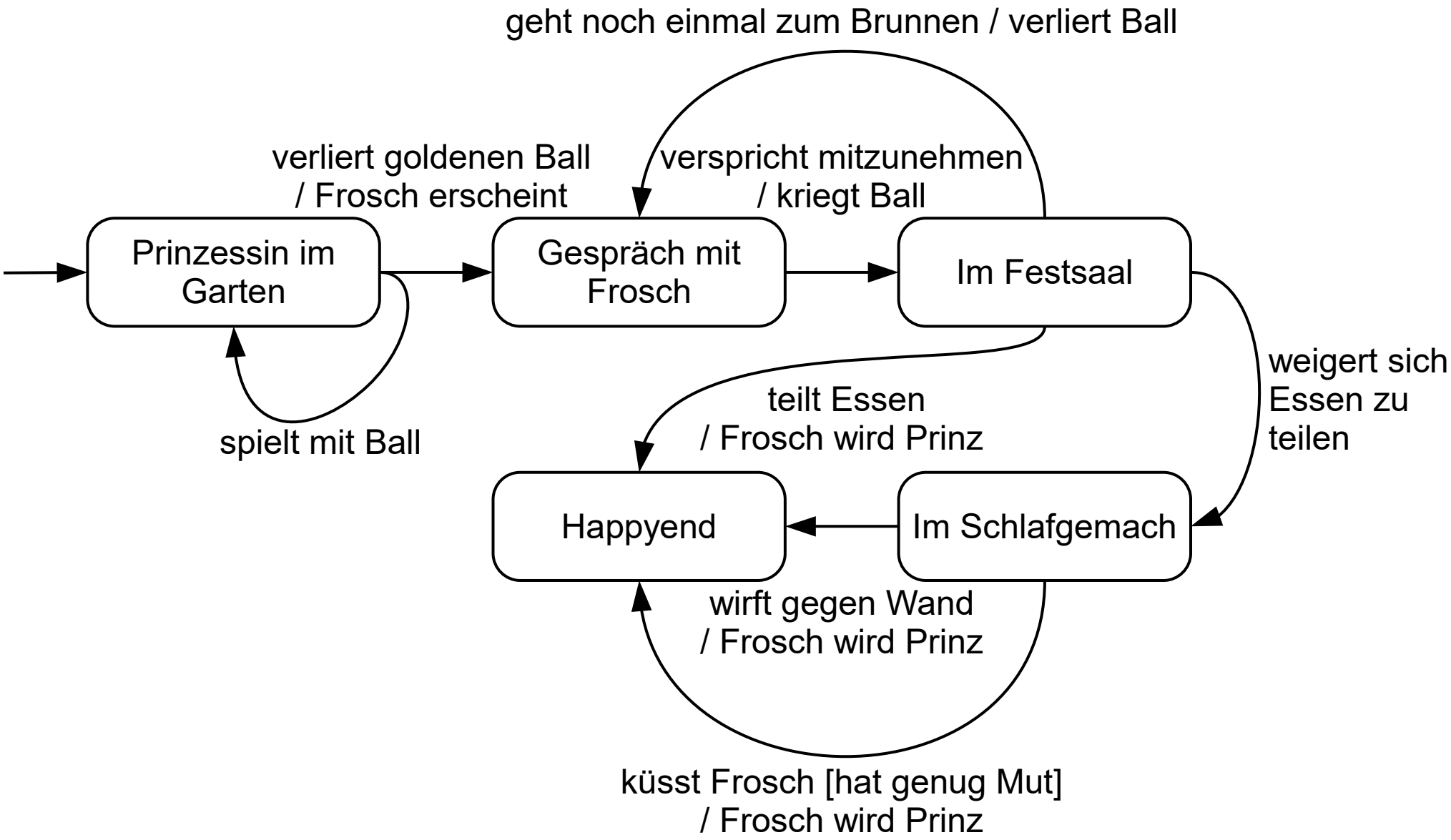
Alter Hut für Deutschlehrer

- “Teile die Geschichte in Abschnitte ein.”



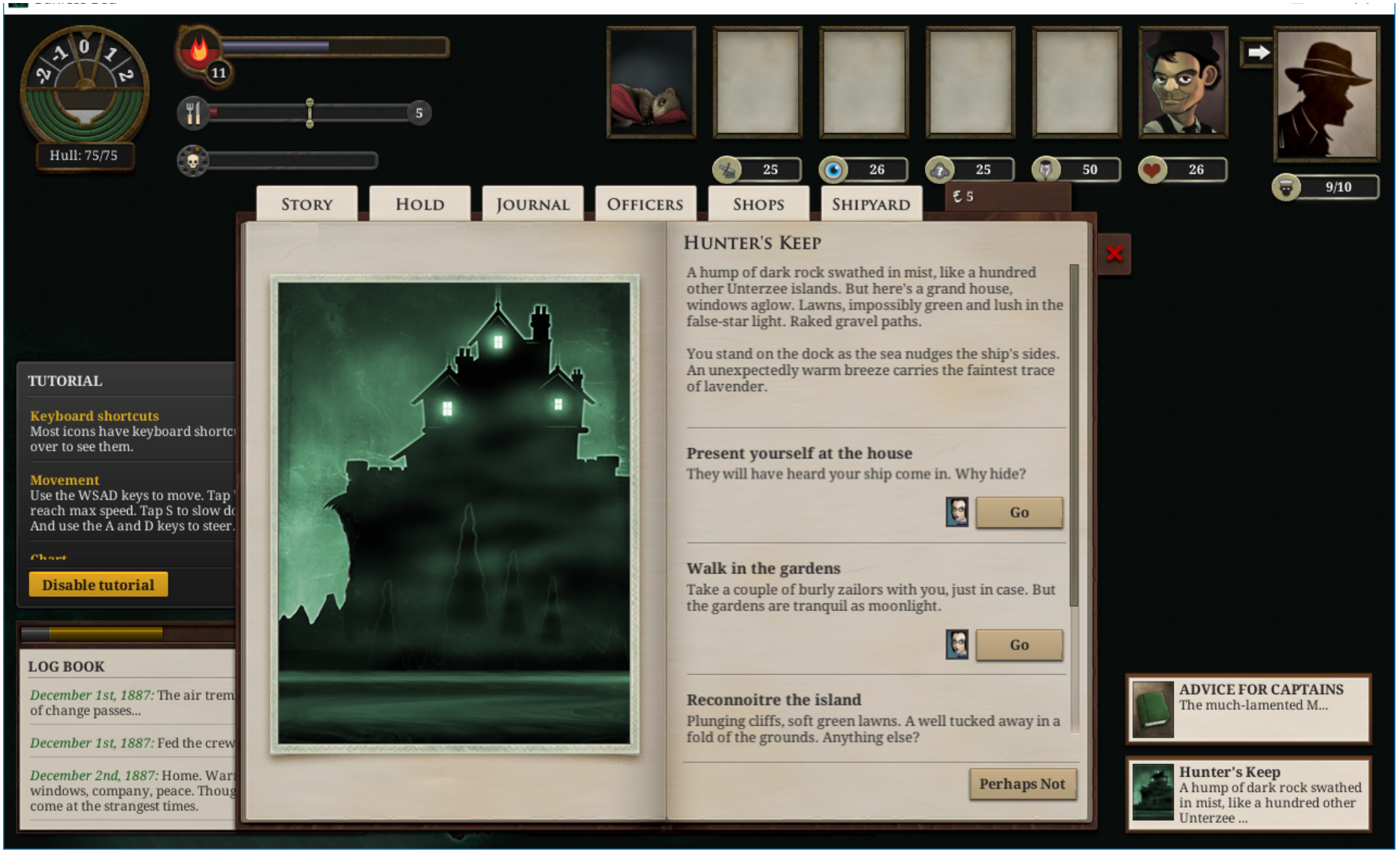
Mehr Möglichkeiten bei (nicht linearen) Computerspielen

Mehr Möglichkeiten bei (nicht linearen) Computerspielen



Das Computerspiel Sunless Sea als Vorbild für modulare Spiele

Sunless Sea



http://www.failbettergames.com/sunless/

Modularer Aufbau

- Downloadable Content: Zusatzmissionen
- Bonus zum Beispiel: Weihnachten 2015

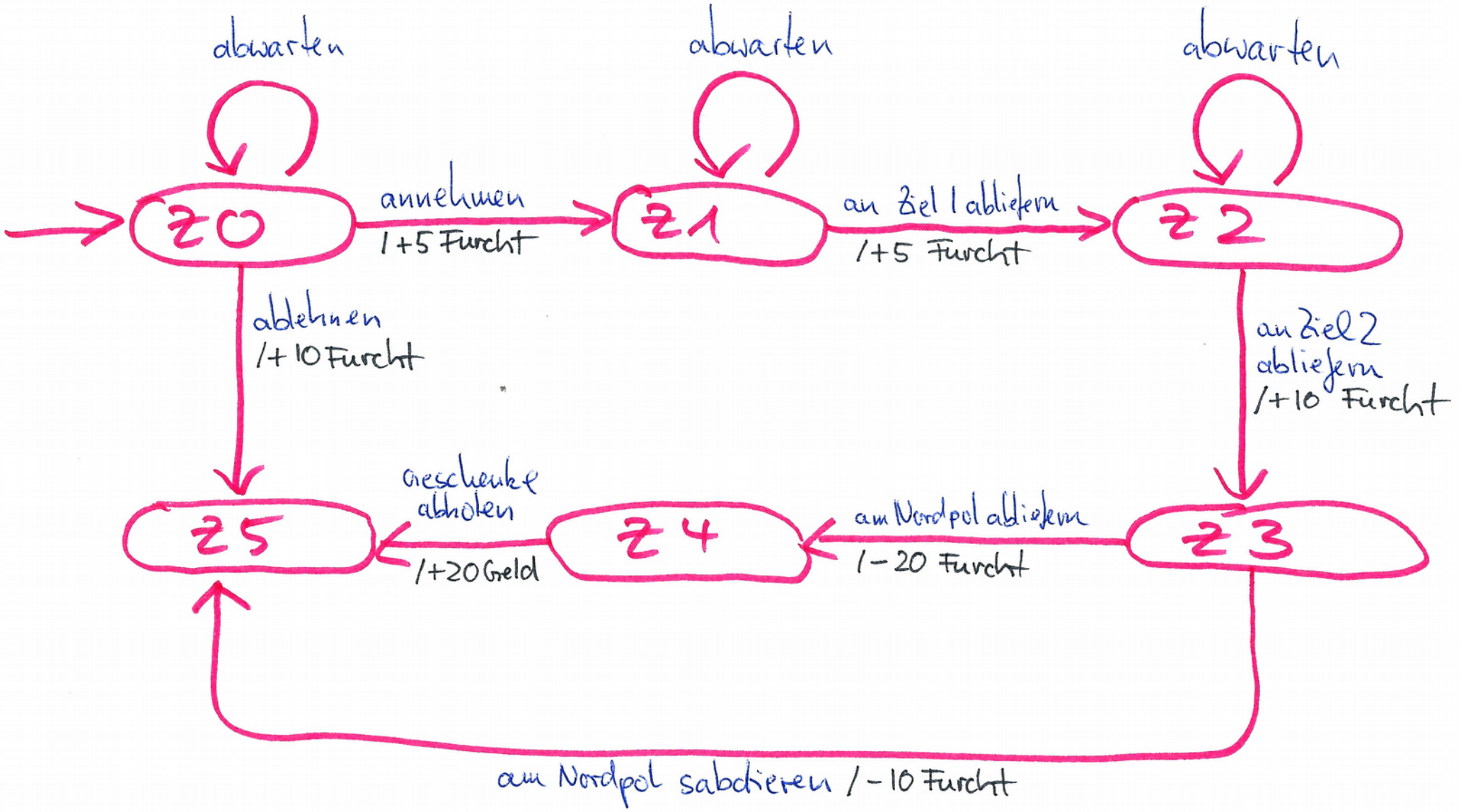
Bildquelle und kurzer Artikel des Modul-Autors Richard Corbbett:
http://www.richardcobbett.com/codex/sunless-sea-and-me-christmastide/

Link zum Thema modulare Spiele:
http://www.gamasutra.com/view/news/262244/Sunless_Sea_80_Days_and_the_rise_of_modular_storytelling.php

Mr Sacks: Eine dunkle Weihnachtsgeschichte

- Mr Sacks verlangt vom Spieler, ihn und 3 finstere Gaben an 3 Orte zu bringen.
- Spieler kann annehmen oder ablehnen (→Mission beendet).
- Nimmt der Spieler an und die Reise dauert zu lange, stirbt der Spieler (→Mission und Spiel beendet).
- Nach jeder Abgabe steigt die Furcht der Spielfigur.
- Beim dritten Ort kann der Spieler die Abgabe sabotieren (→Furcht sinkt, Mission beendet) oder durchführen (→Furcht sinkt), worauf in seiner Wohnung eine Belohnung auf ihn wartet (→Geld steigt, Mission beendet).

Mr Sacks: Eine dunkle Weihnachtsgeschichte



Praxis: Makrosprache

a_wager.ink

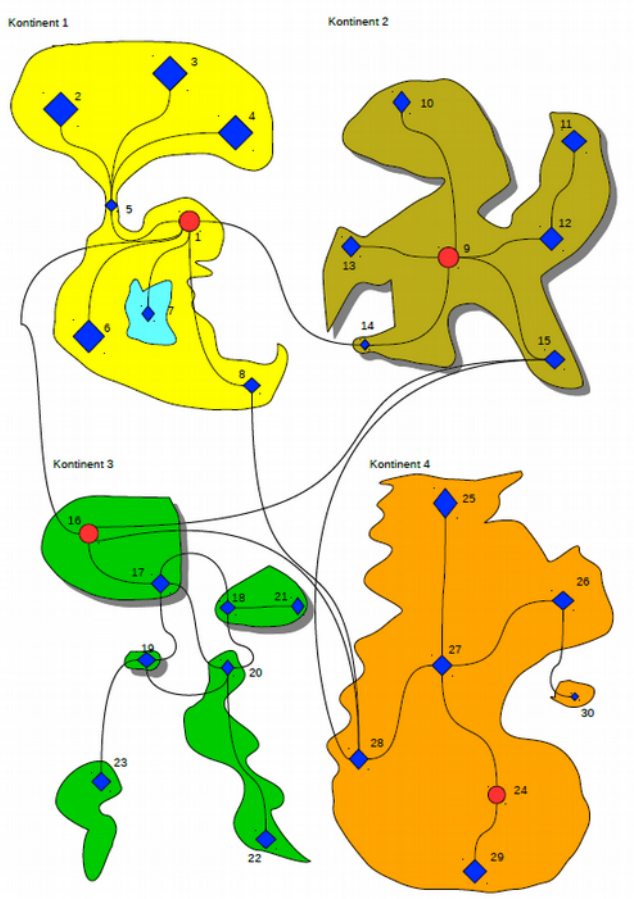
```
1   - I looked at Monsieur Fogg
2   *   ... and I could contain myself no longer.
3       'What is the purpose of our journey, Monsieur?'
4       'A wager,' he replied.
5       * *     'A wager!'[] I returned.
6               He nodded.
7               * * *   'But surely that is foolishness!'
8               * * *  'A most serious matter then!'
9               - - -   He nodded again.
10              * * *   'But can we win?'
11                      'That is what we will endeavour to find out,' he answered.
12              * * *   'A modest wager, I trust?'
13                      'Twenty thousand pounds,' he replied, quite flatly.
14              * * *   I asked nothing further of him then[.], and after a final,
                    polite cough, he offered nothing more to me. <>
15      * *     'Ah[.'],' I replied, uncertain what I thought.
16      - -     After that, <>
17  *   ... but I said nothing[] and <>
18  - we passed the day in silence.
19  - -> END
```

110 Words, Line 3, Column 52 Spaces: 4 ink

Ergebnis des Klassenprojekts

Ergebnis des Klassenprojekts

- Erstellen einer gemeinsamen Welt mit Orten und Verbindungen (je Schülerin ein Ort)
- Jede Schülerin legt eine Geschichte an, die an mindestens einem der Orte spielt.

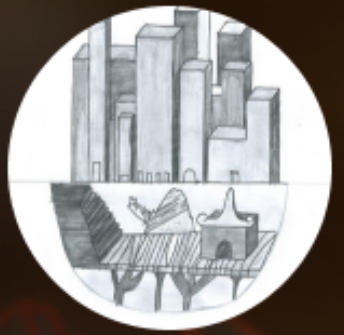


Rostwelt

Rostwelt

GroßStaat | Spielhalle

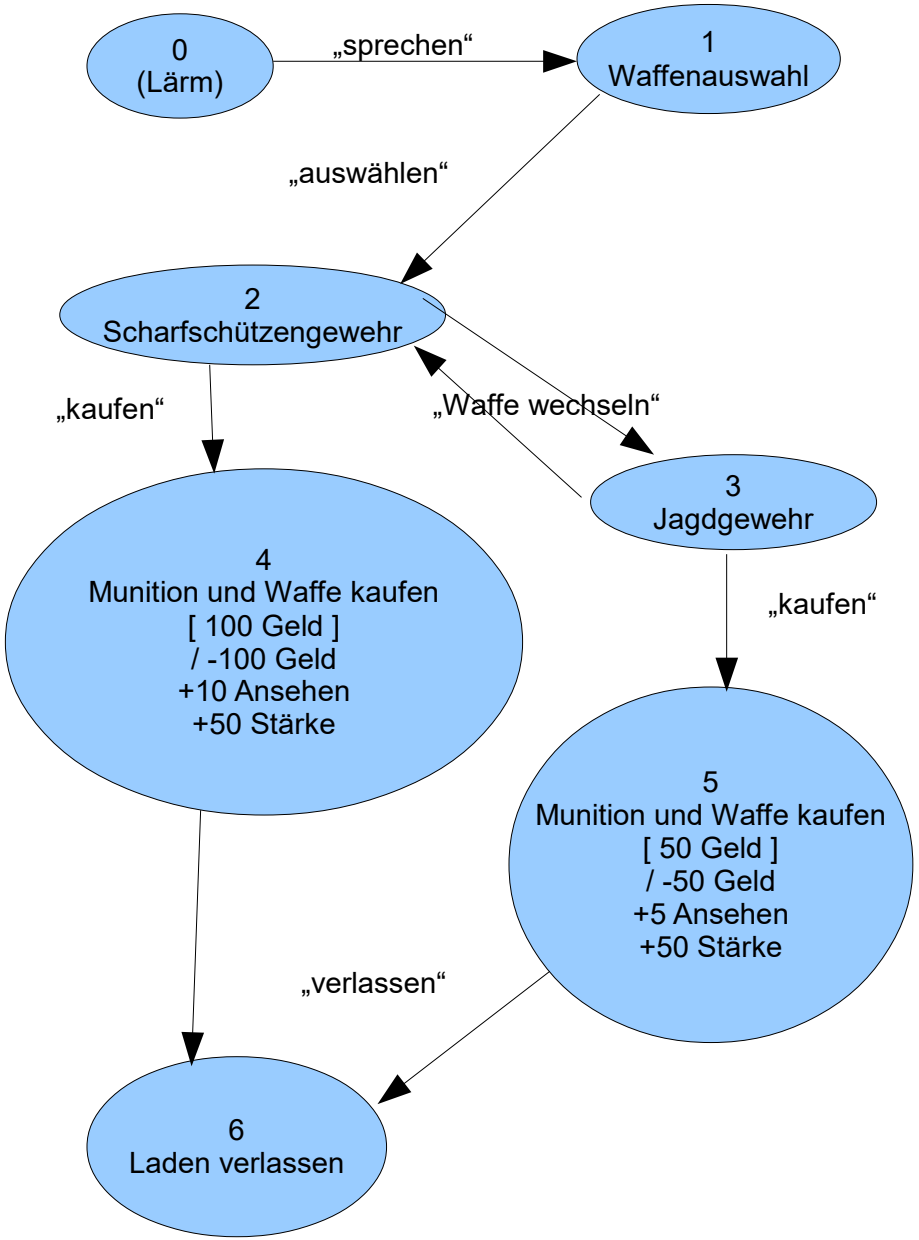
GroßStaat



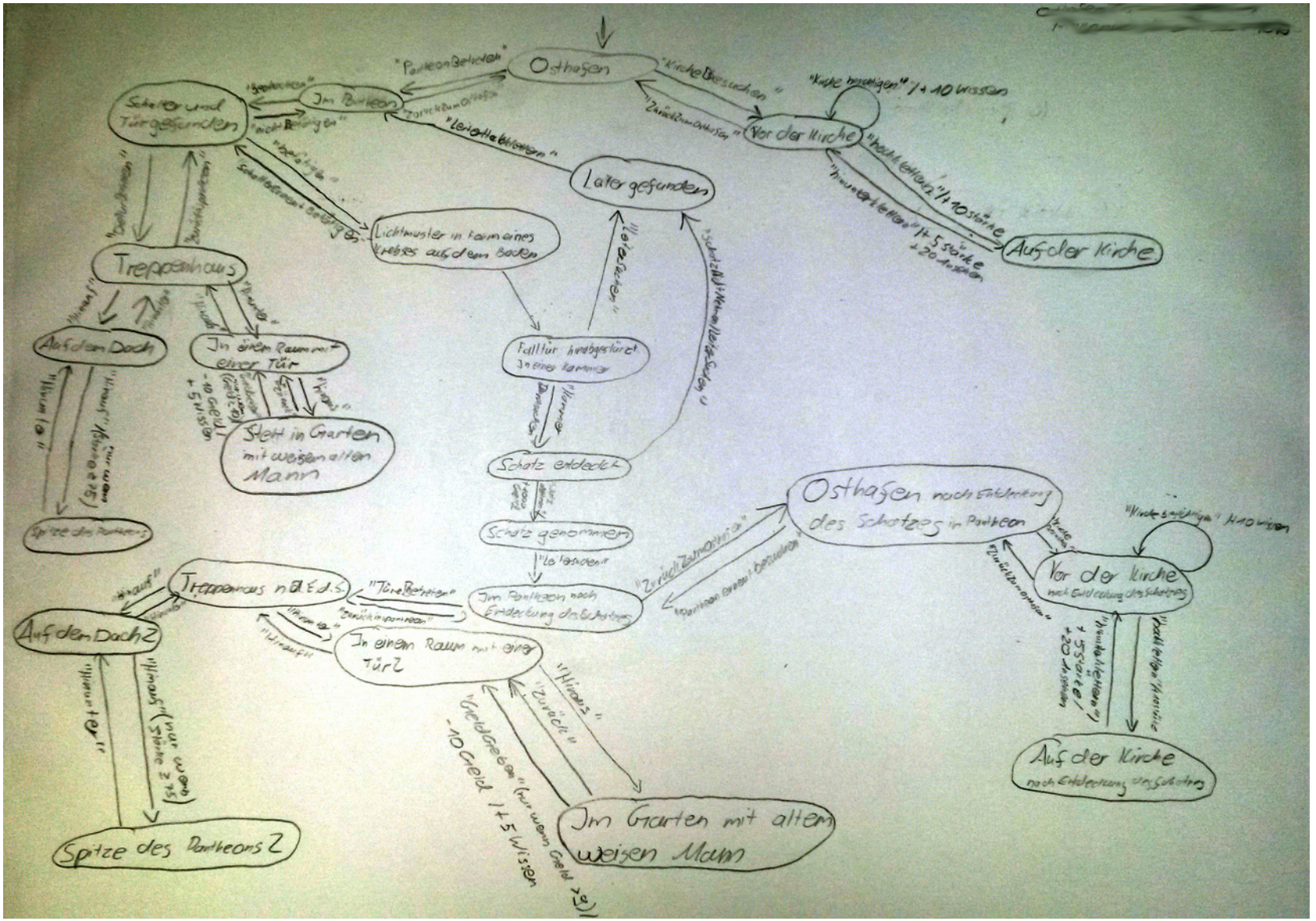
Eine riesige, beinahe rostfreie Metropole, die nicht nur für ihre Bauwerke, sondern auch für ihre wichtige Verbindung zwischen dem Brückenpunkt und den südlichen Städten bekannt ist.

Brückenpunkt | Verlassene Bibliothek | Rostsee | Südriff | Innismund | Werft

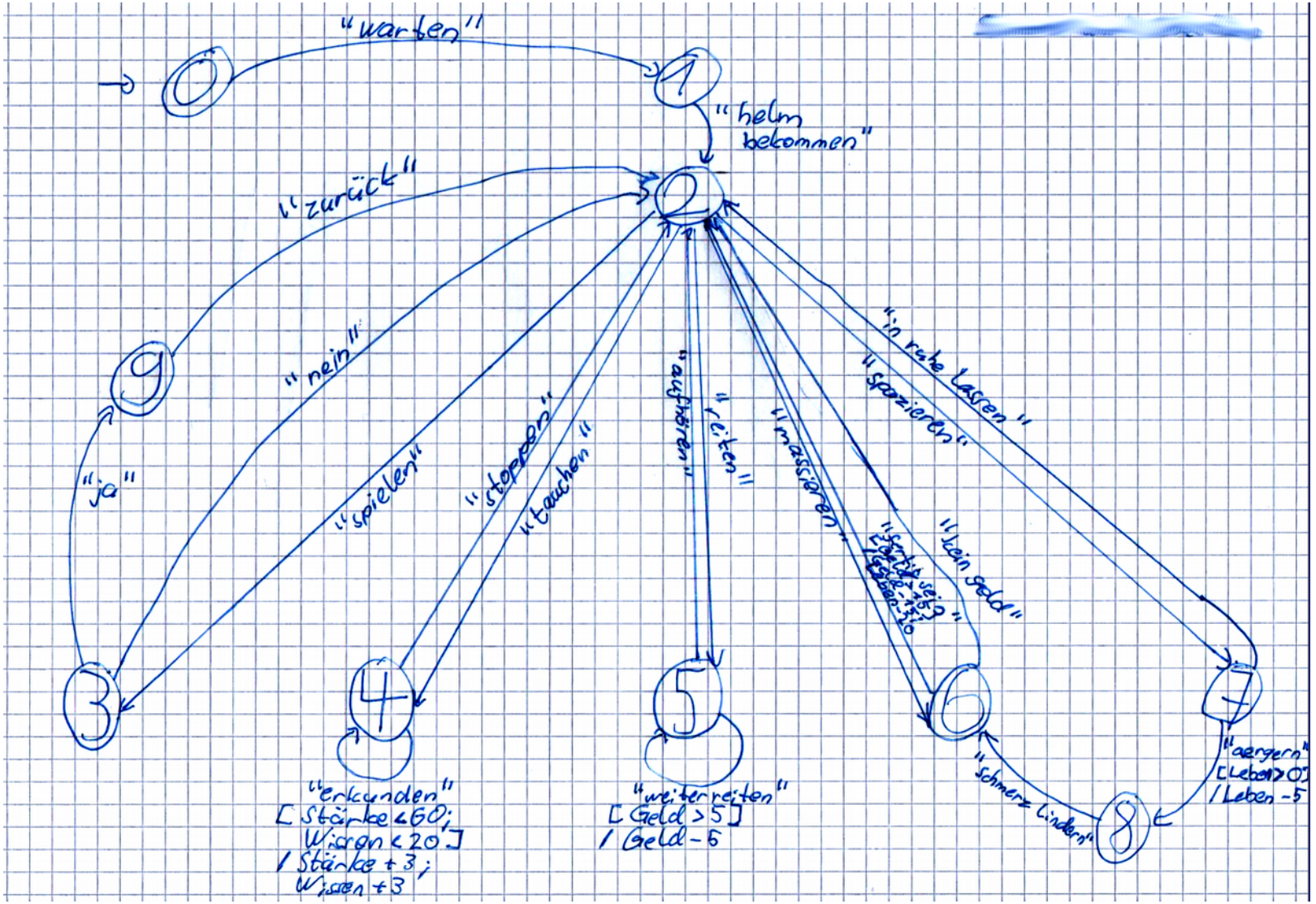
Automat "Jagdgebrauch"



Automat "Osthafen"



Automat "Oase"



Oase

Die Oase ist ein Ort der Erholung, der Freizeit und des Vergnügens.

Beim Betreten der Oase wird man zuerst von einem Touristenguide eingewiesen und man bekommt einen Helm zum Schutz vor Kokosnüssen welche von den Bäumen fallen könnten.

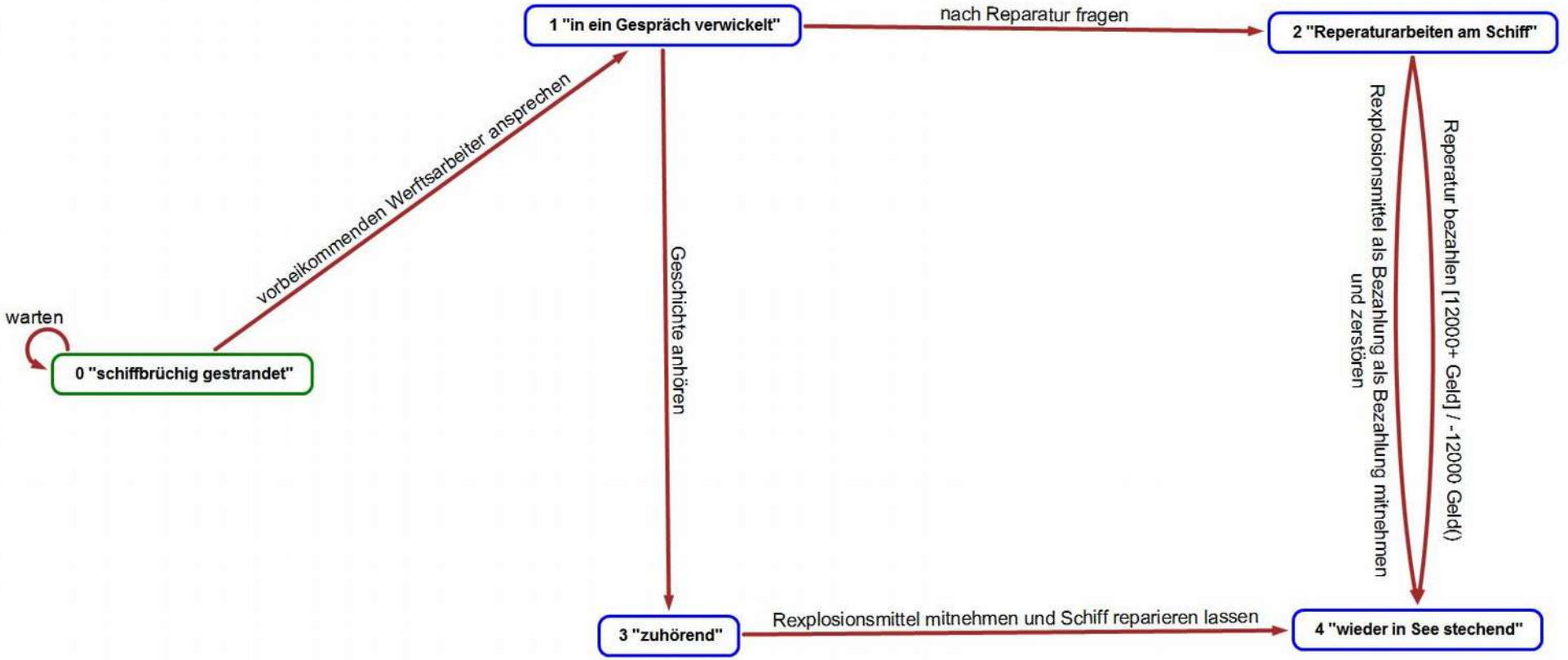
Nun gibt es verschiedene Möglichkeiten an Aktivitäten, welche man ausführen kann.

In naher Zukunft soll es ein Memory geben, welches als Fun- Faktor dient. Außerdem kann man Tauchen gehen, auf Elefanten reiten oder eine Massage genießen, welche von Elefanten ausgeübt wird. Einfach spazieren gehen, die Flora und Fauna genießen, ist auch möglich. Hierbei ist jedoch ist jedoch auf die Affen Acht zu geben, welche auf den Kokospalmen leben. Denn wenn sie sich gestört oder geärgert fühlen, werfen sie schnell mit Kokosnüssen auf die Oasenbesucher.

Zum Schutz vor den Kokosnüssen dient der Helm, den der Tourist beim Betreten der Oase geschenkt bekommen hat. Falls er von einer Kokosnuss getroffen wird, begibt sich der Tourist zur Massage, zur Entspannung und Regeneration.

Doch was dort passiert, hätte man nicht erwartet…

Automat "Pontieruga"



Versorgungshafen Pontiéruga

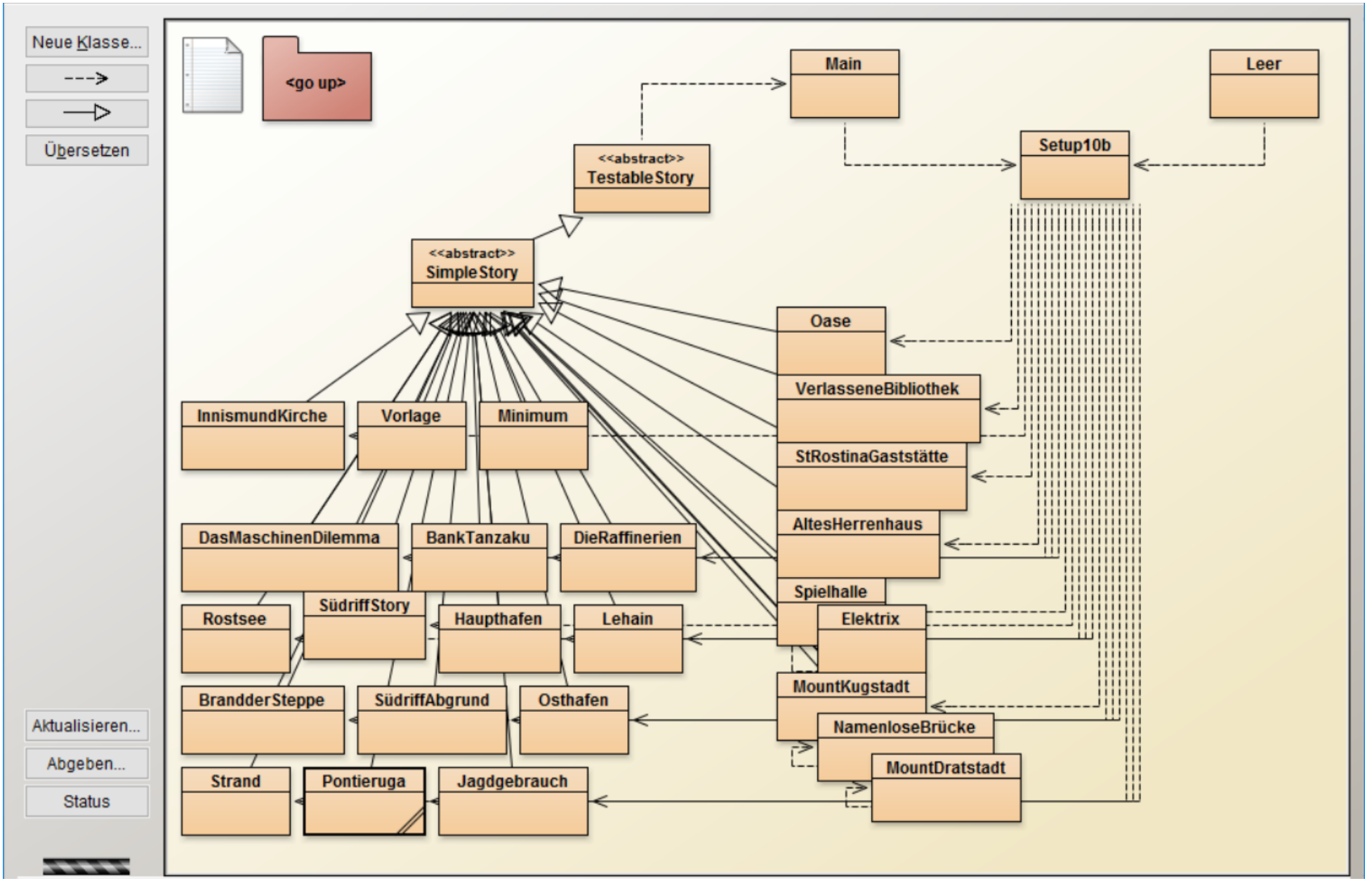
Der Ausgangspunkt der Geschichte ist die schiffbrüchige Strandung auf der Insel. Das Schiff ist reparaturbedüftig, weswegen man eine Möglichkeit finden muss, es wieder in Gang zu setzen, um weitersegeln zu können. Beim Antreffen eines Werftsarbeiters hat man die Auswahl zwischen der Frage nach einer Reparatur und dem Anhören einer Geschichte, um mehr über Land und Leute zu erfahren.

Bittet man um Hilfe bei der Instandsetzung des Boots, ist es von Nöten, eine hohe (unbezahlbare) Summe von 12000 Geld bereitzustellen. Andernfalls muss man ein gefährliches Rexplosionsmittel entsorgen. Das Ziel ist es, dass die Option des Zahlens nicht infrage kommt und es somit immer auf die Entsorgung der Flüssigkeit hinausläuft.

Dies geschieht auch, indem man sich die Erzählung anhört, da sie von ebendiesem Rexplosionsmittel handelt. Daraufhin bietet man sich von selbst an, es zu entsorgen und das Schiff wird aus Dankbarkeit repariert.

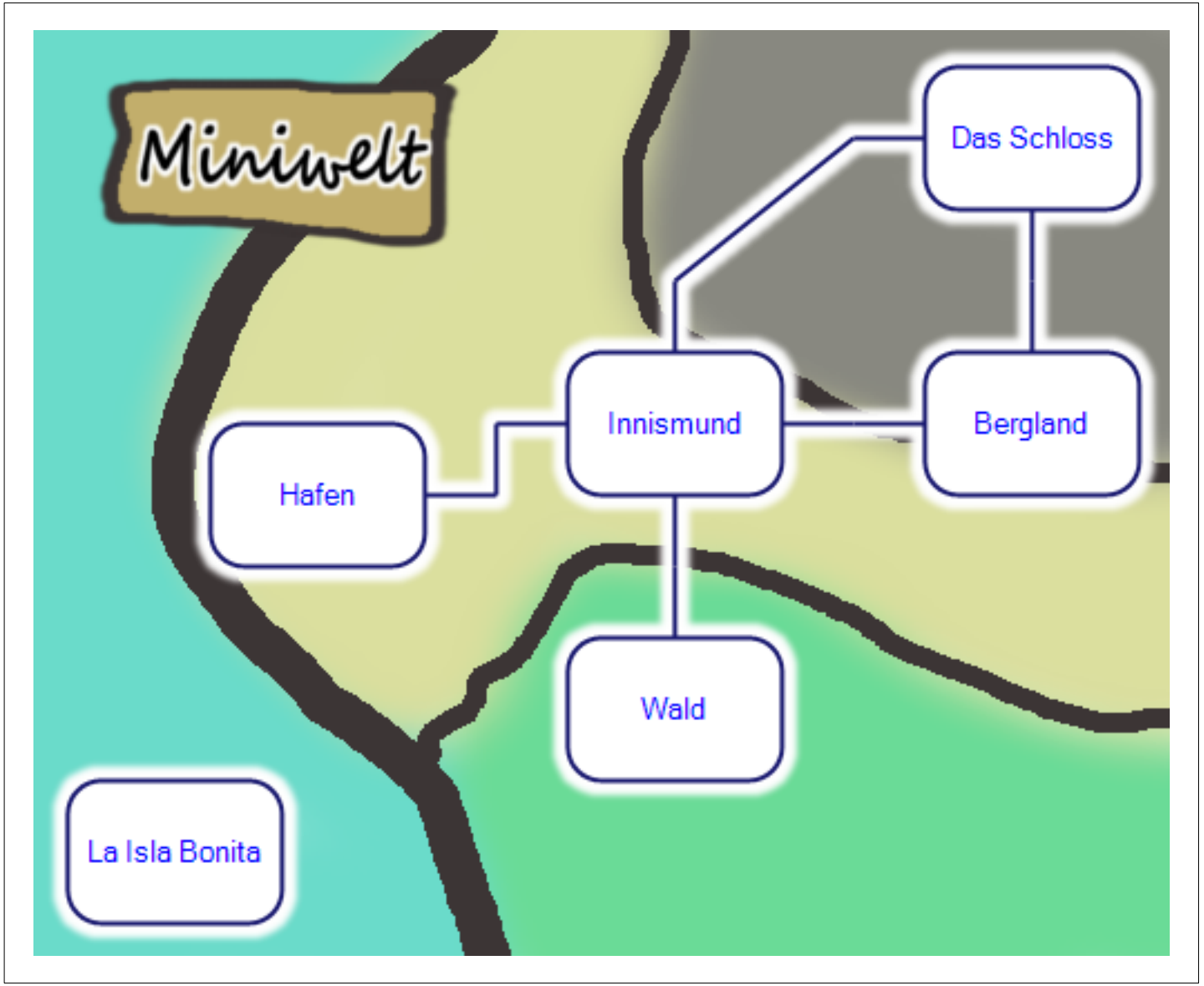
Insgesamt werden sowohl die Ziele des Spielers (Schiff reparieren, Insel verlassen), als auch die Ziele der Inselbewohner (Rexplosionsmittel loswerden) erreicht.

BlueJ


Neue Klasse...
Übersetzen
Aktualisieren...
Abgeben...
Status
<go up>
Main
Leer
Setup10b
<<abstract>>
TestableStory
<<abstract>>
SimpleStory
Oase
VerlasseneBibliothek
StRostinaGaststätte
AltesHerrenhaus
Spielhalle
Elektrix
MountKugstadt
NamenloseBrücke
MountDratstadt
InnismundKirche
Vorlage
Minimum
DasMaschinenDilemma
BankTanzaku
DieRaffinerien
Rostsee
SüdriffStory
Haupthafen
Lehain
BrandderSteppe
SüdriffAbgrund
Osthafen
Strand
Pontieruga
Jagdgebrauch

Selbermachen in der Fortbildung

Die Fortbildungswelt



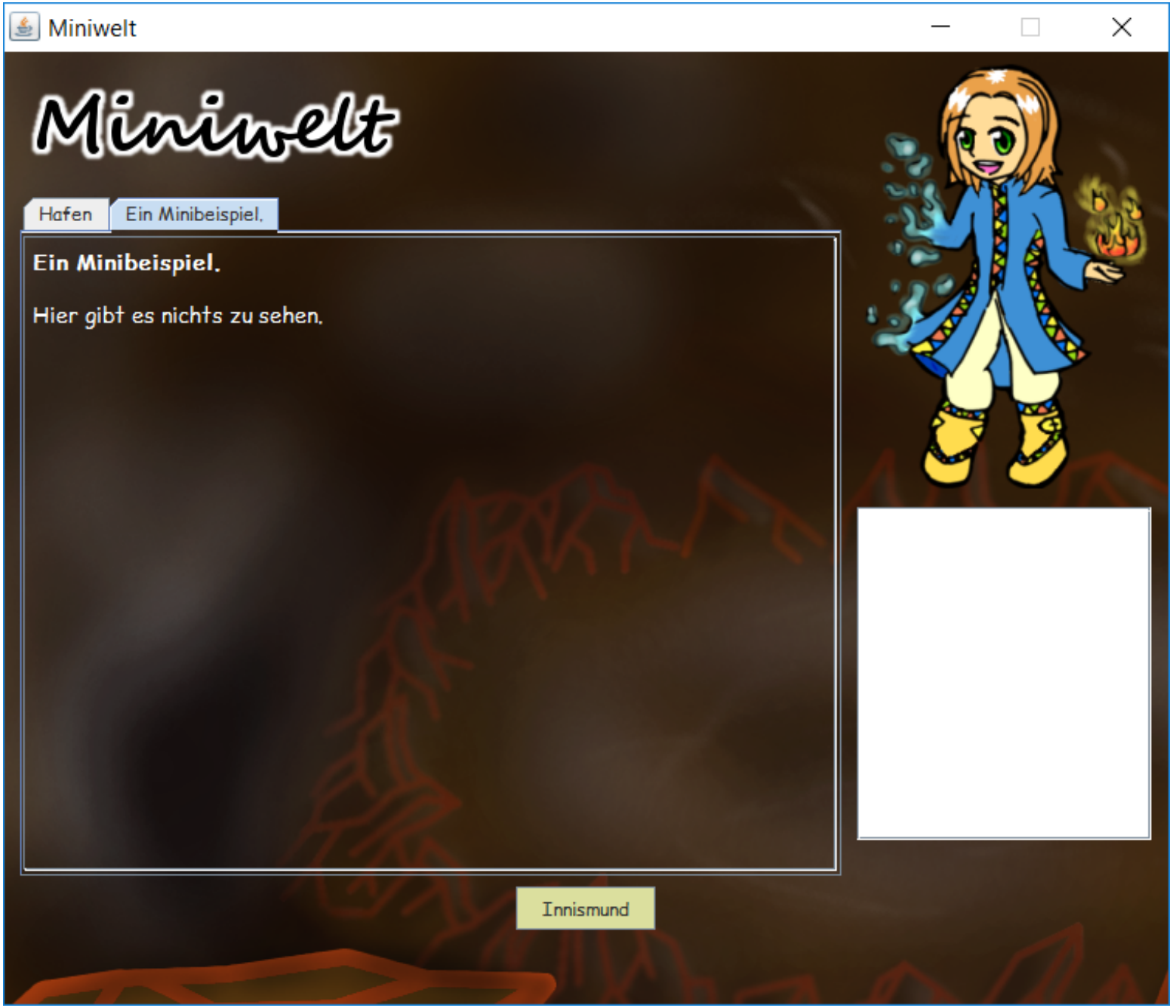
- Öffnen des BlueJ-Projekts
- Arbeitsblatt 1: Die kleinste mögliche Geschichte
- Arbeitsblatt 2: Eine Geschichte mit Optionen
- Arbeitsblatt 3: Bessere Optionen
- Arbeitsblatt 4: Umsetzung eines Automaten
- Arbeitsblatt 5: Geschichten mit mehreren Orten
- Optional: Abgabe und Zusammenbau

Arbeitsblatt 1: Anlegen einer eigenen Klasse (kopieren und umbenennen der Vorlage Arbeitsblatt1)

```
import storyworld.*;
public class Arbeitsblatt1 extends SimpleStory {
 public Arbeitsblatt1() { test(); }
 public String getName() { return "Beispiel."; }
 public String startStoryAt() { return "Hafen"; }
 public Report createReport() {
  Report r = new Report("Nichts zu sehen.");
  return r;
 }
}
```

Ideen: Baum im Wald, Vampir im Schloss, Rathaus in Innismund

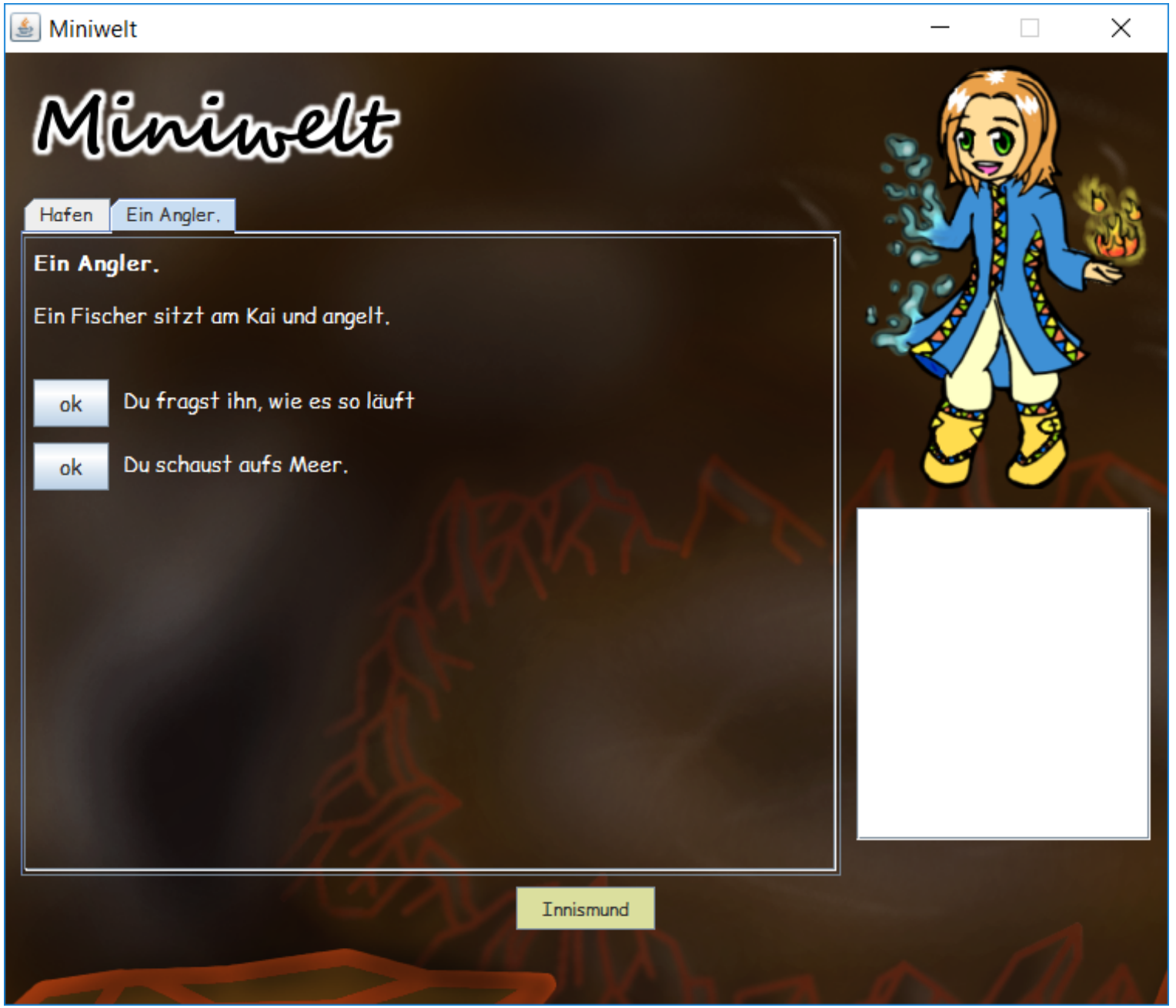
Arbeitsblatt 1: Ergebnis



Arbeitsblatt 2: Hinzufügen von Optionen

```
import storyworld.*;
public class Arbeitsblatt2 extends SimpleStory {
 public Arbeitsblatt2() { test(); }
 public String getName() { return "Ein Angler"; }
 public String startStoryAt() { return "Hafen"; }
 public Report createReport() {
  Report r = new Report("Ein Fischer sitzt da.");
  r.addOption("Du fragst ihn, wie...", "fragen");
  r.addOption("Du schaust aufs Meer.", "schauen");
  return r;
 }
}
```

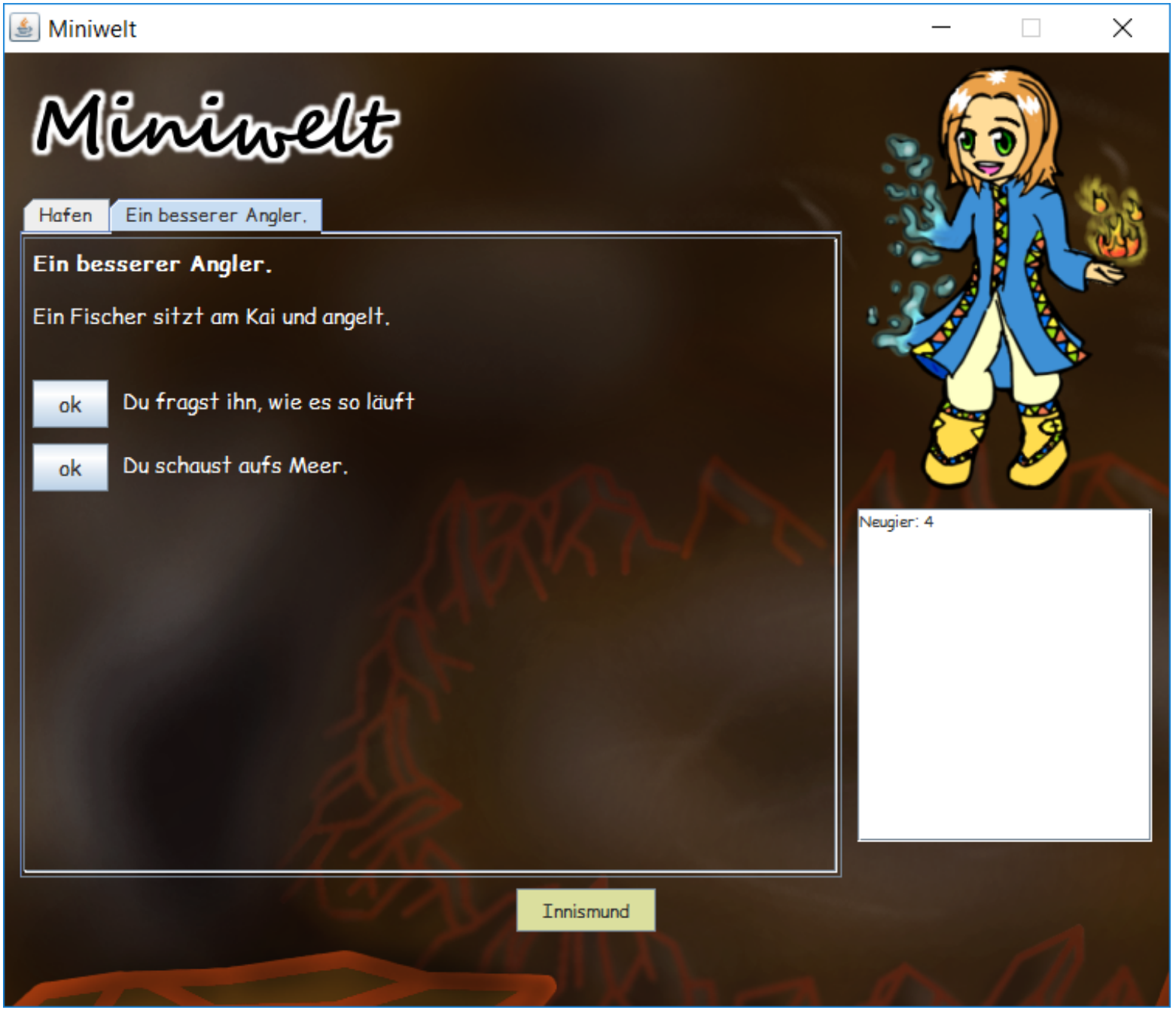
Arbeitsblatt 2: Ergebnis



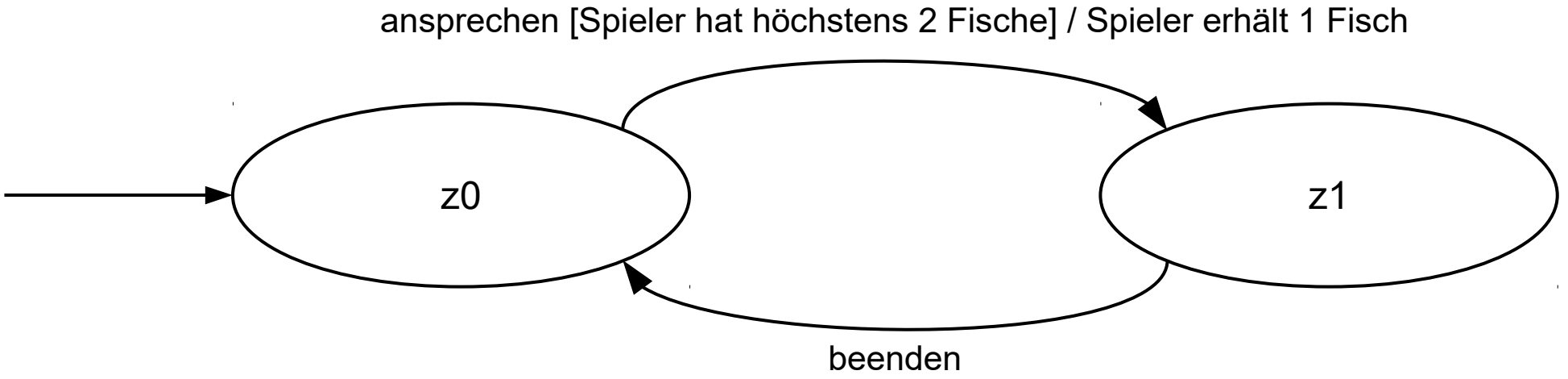
Arbeitsblatt 3: Auswirkung von Optionen

```
import storyworld.*;
public class Arbeitsblatt3 extends SimpleStory {
 public Arbeitsblatt3() { test(); }
 public String getName() { return "Ein Angler"; }
 public String startStoryAt() { return "Hafen"; }
 public Report createReport() {
  Report r = new Report("Ein Fischer sitzt da.");
  r.addOption("Du fragst ihn, wie...", "fragen");
  r.addOption("Du schaust aufs Meer.", "schauen");
  return r;
 }
  public void receiveMessage(String nachricht) {
   if (nachricht.equals("fragen")) {
    increase("Neugier", 1);
   }
   else if (nachricht.equals("schauen")) {
    set("Innerer Frieden", 20);
   }
  }
}
```

Arbeitsblatt 3: Ergebnis



Arbeitsblatt 4a: Bessere Rückmeldung mit Zuständen



- z0: Fischer sitzt da und angelt.
  - z0, ansprechen (wenn Spieler max. 2 Fische hat)
    > Übergang zu z1, wobei Spieler 1 Fisch erhält
- z1: Fischer freut sich über Ansprache.
  - z1, beenden des Gesprächs
    > Übergang zu z0
- abgegebener Report hängt von Zustand ab

Arbeitsblatt 4a: Bessere Rückmeldung mit Zuständen

```
public Report createReport() {
 if (state == 0) {
  Report r = new Report("Ein Fischer sitzt am Kai und angelt.");
  r.addOption("Du fragst ihn, wie es so läuft.", "ansprechen");
  return r;
 } else {
  Report r = new Report("Der Angler erzählt dir eine Geschichte und gibt dir einen Fisch.");
  r.addOption("Du dankst für das Gespräch und den Fisch.", "beenden");
  return r;
 }
}
public void receiveMessage(String s) {
 if (state==0) {
  if (s.equals("ansprechen") ) // ausloesende Aktion
  {
   if (get("Fische")<3) { // Bedingung des Zustandsuebergangs
    state = 1;
    increase("Fische", 1); // ausgeloeste Aktionen
   } else {
    sendMessage("Du hast schon genug Fische.");
   }
  }
 }
 else if (state==1) {
  if (s.equals("beenden")) // ausloesende Aktion
  {
   state = 0;
  }
 }
}
```

Arbeitsblatt 4a: Bessere Rückmeldung mit Zuständen
Änderungen bei der createReport()-Methode

```
public Report createReport() {
 if (state == 0) {
  Report r = new Report("Ein Fischer sitzt am Kai und angelt.");
  r.addOption("Du fragst ihn, wie es so läuft.", "ansprechen");
  return r;
 } else {
  Report r = new Report("Der Angler gibt dir einen Fisch.");
  r.addOption("Du dankst für den Fisch.", "beenden");
  return r;
 }
}
```

Arbeitsblatt 4a: Bessere Rückmeldung mit Zuständen
Änderungen bei der receiveMessage(String)-Methode

```
public void receiveMessage(String s) {
 if (state==0) {
  if (s.equals("ansprechen") ) // ausloesende Aktion
  {
   if (get("Fische")<3) { // Bedingung des Zustandsuebergangs
    state = 1;
    increase("Fische", 1); // ausgeloeste Aktionen
   } else {
    sendMessage("Du hast schon genug Fische.");
   }
  }
 }
 else if (state==1) {
  if (s.equals("beenden")) // ausloesende Aktion
  {
   state = 0;
  }
 }
}
```

Arbeitsblatt 4b: Alternative Lösung

- Erbt von Klasse SimpleStoryCG
- *Keine* Methode receiveMessage(String)
- Stattdessen werden die mit der Option assoziiertern Strings als Bezeichner von Methoden interpretiert, die dann ausgeführt werden

```
public Report createReport() {
  …
  r.addOption("Du fragst ihn, wie es so
                          läuft.", "ansprechen");
….
}

public void ansprechen() {
  if (get("Fische")<3) {
    state = 1;
    increase("Fische", 1);
  } else {
    sendMessage("Du hast genug.");
  }
}
```

Arbeitsblatt 5: Geschichten an mehreren Orten

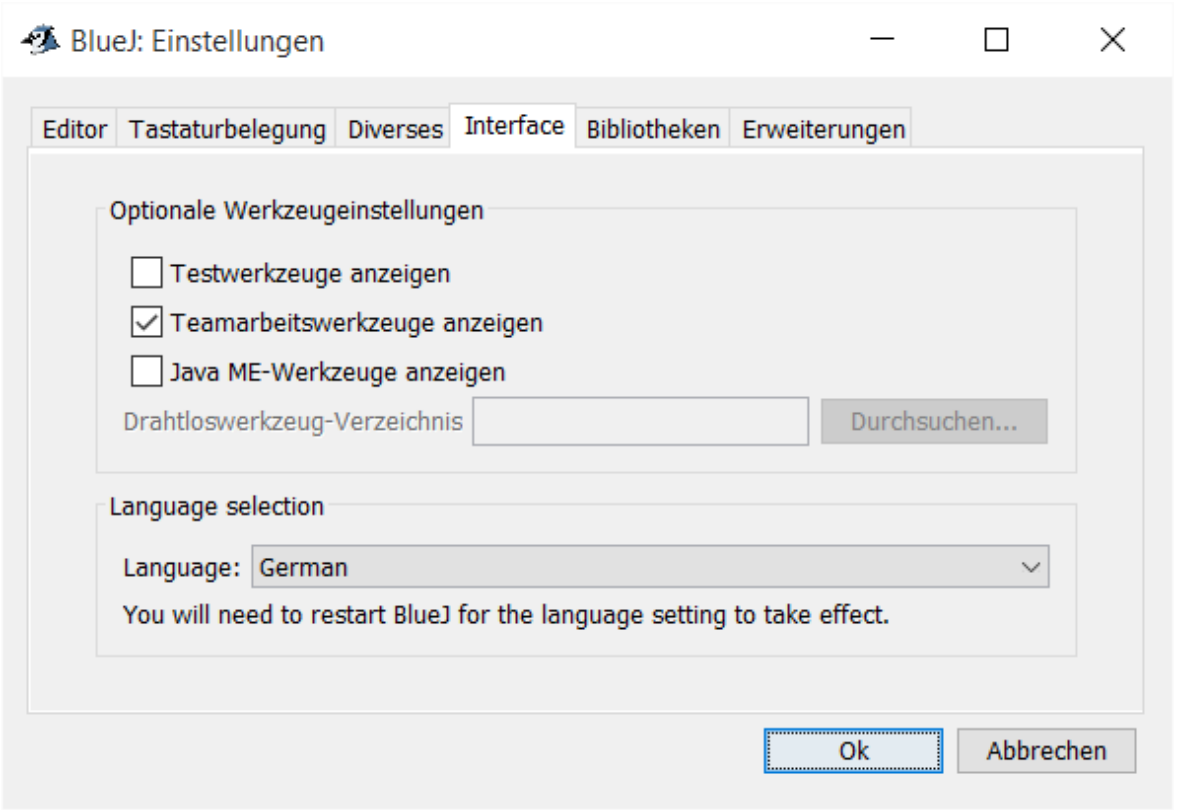
```
import storyworld.*;
public class Arbeitsblatt5 extends MultiplaceStory {
 String imageName;
 public Arbeitsblatt5() { test();}
  public String startStoryAt() { return "Das Schloss"; }
  public String getName() { return "Romeo & Julia"; }
  public Report createReport() {
   if (playerIsIn("Das Schloss")) {
    imageName = "resources/maedchen.png";
    Report r = new Report("Hier sitzt Julia und wartet auf ihren Romeo.");
    return r;
   }
   else if (playerIsIn("Bergland")) {
    imageName = "resources/junge.png";
    Report r = new Report("Hier sitzt Romeo und wartet auf seine Julia.");
    return r;
   }
   else return null;
  }
 public void receiveMessage(String s) {  }
 public String getImageName() { return imageName; }
}
```

Zusammenbau

1. Anzeigen der Teamworkeinstellungen
2. Eingabe des Passworts
3. Hochladen der vorgenommenen Änderungen (Serverdaten sind bereits gespeichert) – eventuell Schwierigkeiten
4. Hinzufügen aller neuen Klassen zur Welt
5. Akualisieren des eigenen Projekts

1. Teamarbeitseinstellungen freischalten

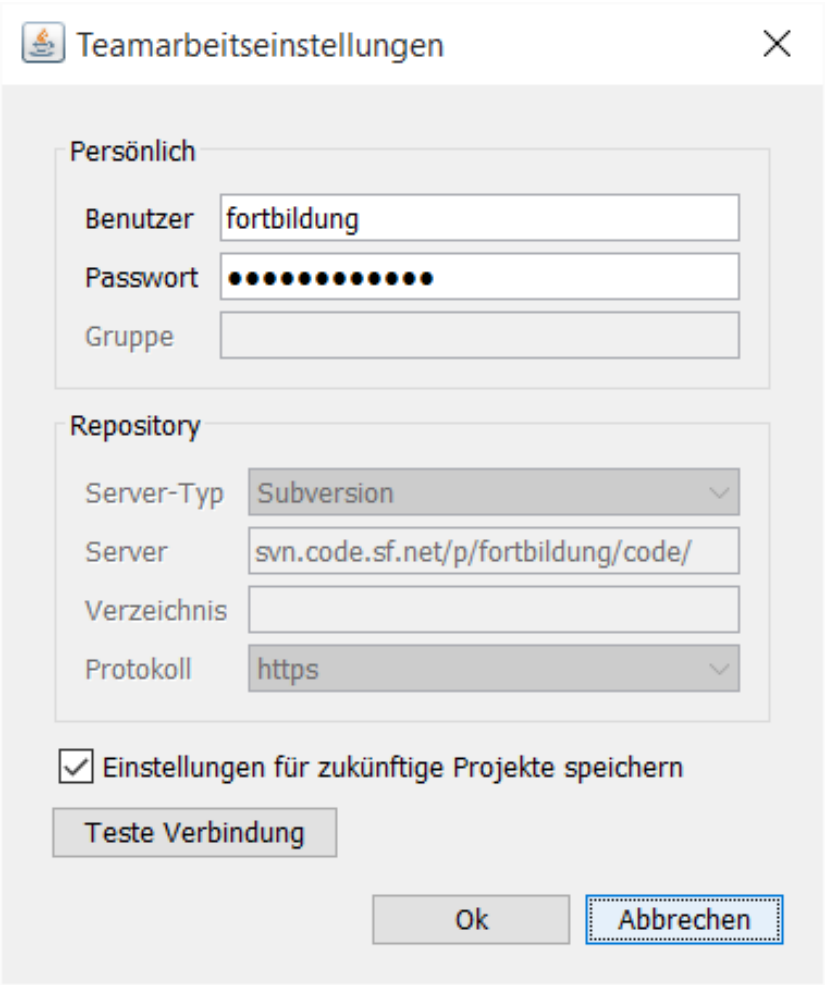
- Menüpunkt Werkzeuge\Einstellungen\Interface: "Teamarbeitswerkzeuge anzeigen" ankreuzen.
  (je nach BlueJ-Version und -Sprache auch anders)



2. Eingabe des Passworts

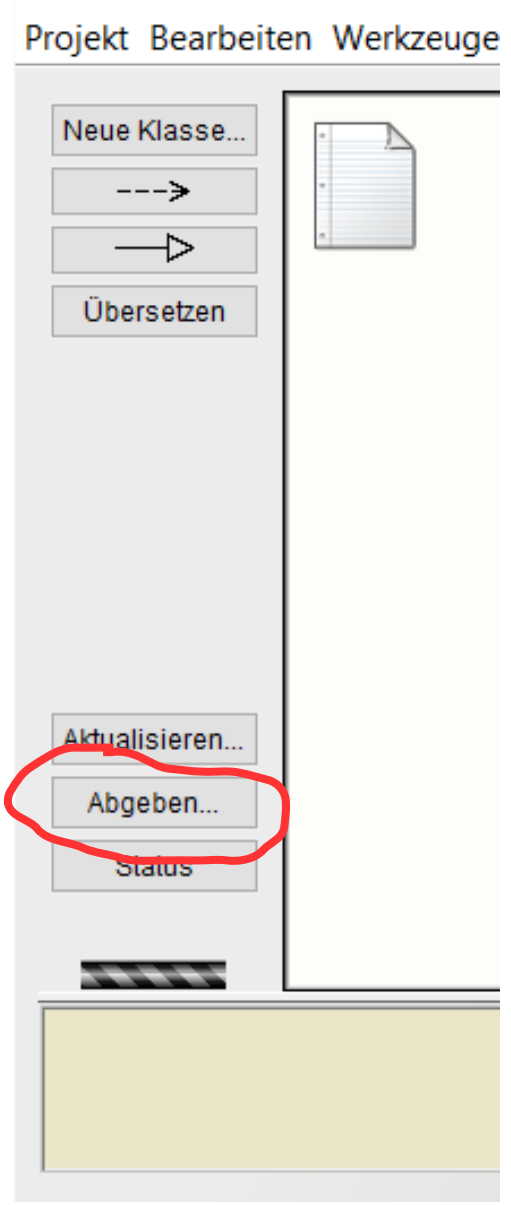
- Menüpunkt Werkzeuge\Teamarbeit\Teamarbeitseinstellungen...

Benutzer: fortbildung
Passwort: ********



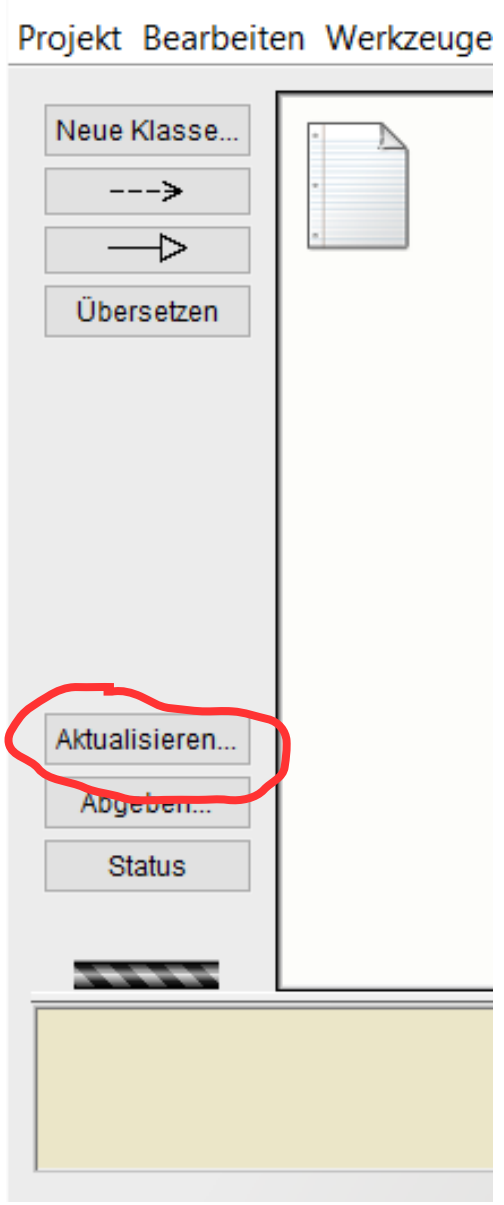
3. Hochladen der eigenen Änderungen

- Auf den neuen Knopf "Abgeben..." klicken und dann immer wieder bestätigen
- Vorsicht: *Alle* Änderungen werden zum Server hochgeladen
- Bei Fehler eventuell Passwort neu eingeben
- Eventuell vorheriges Aktualisieren nötig (automatisch durchgeführt, kann zu Konflikten führen)



5. Aktualisieren der Server-Änderungen

- Auf den neuen Knopf "Aktualisieren..." klicken
- Änderungen am Server werden in das Projekt übertragen



Was bringt Ihnen das?

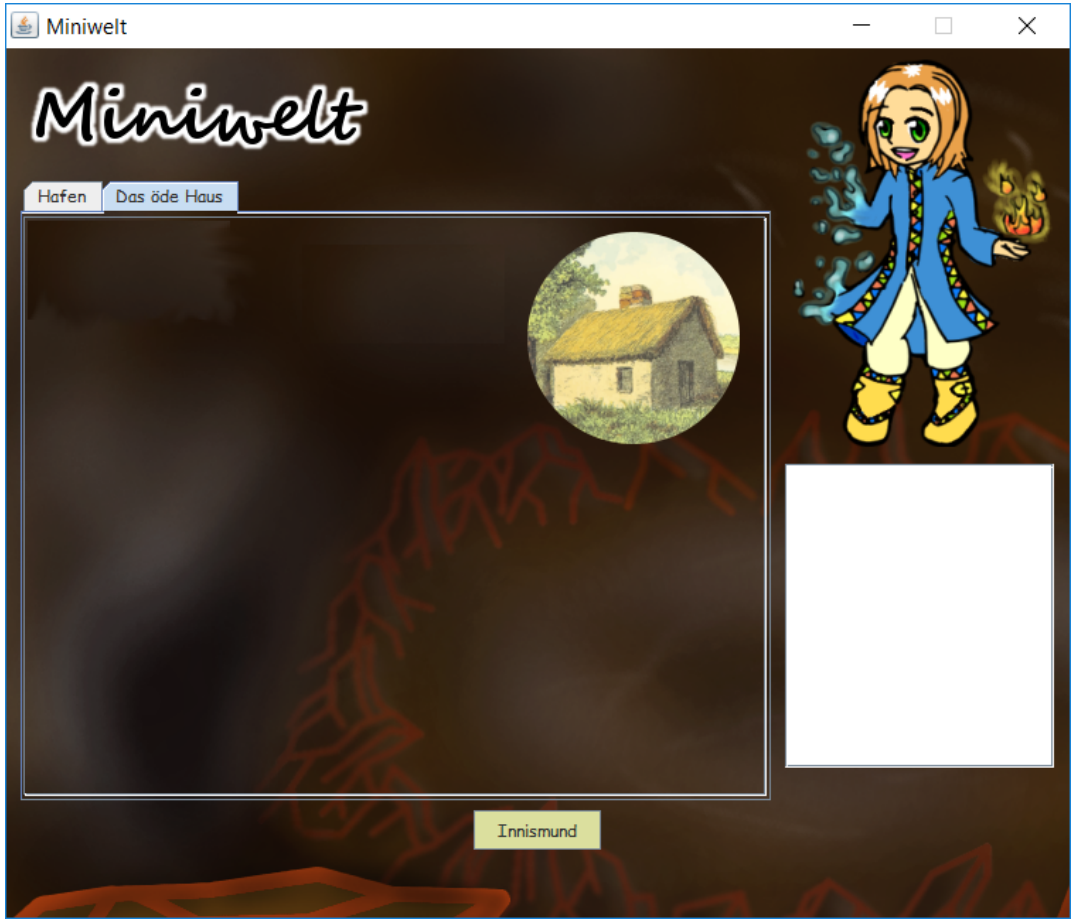
- Für Sie:
  - Anregung
  - Information
  - Code zum Nachmachen

- Vorteile:
  - Beliebiger Schwierigkeitsgrad
  - Teamwork
  - Einsatz von Zustands-Automaten
  - Kreativität, Gestaltung
  - Zusammenarbeit mit Deutsch

Technisches

Zusammenspiel von createReport( ) und receiveMessage(String)

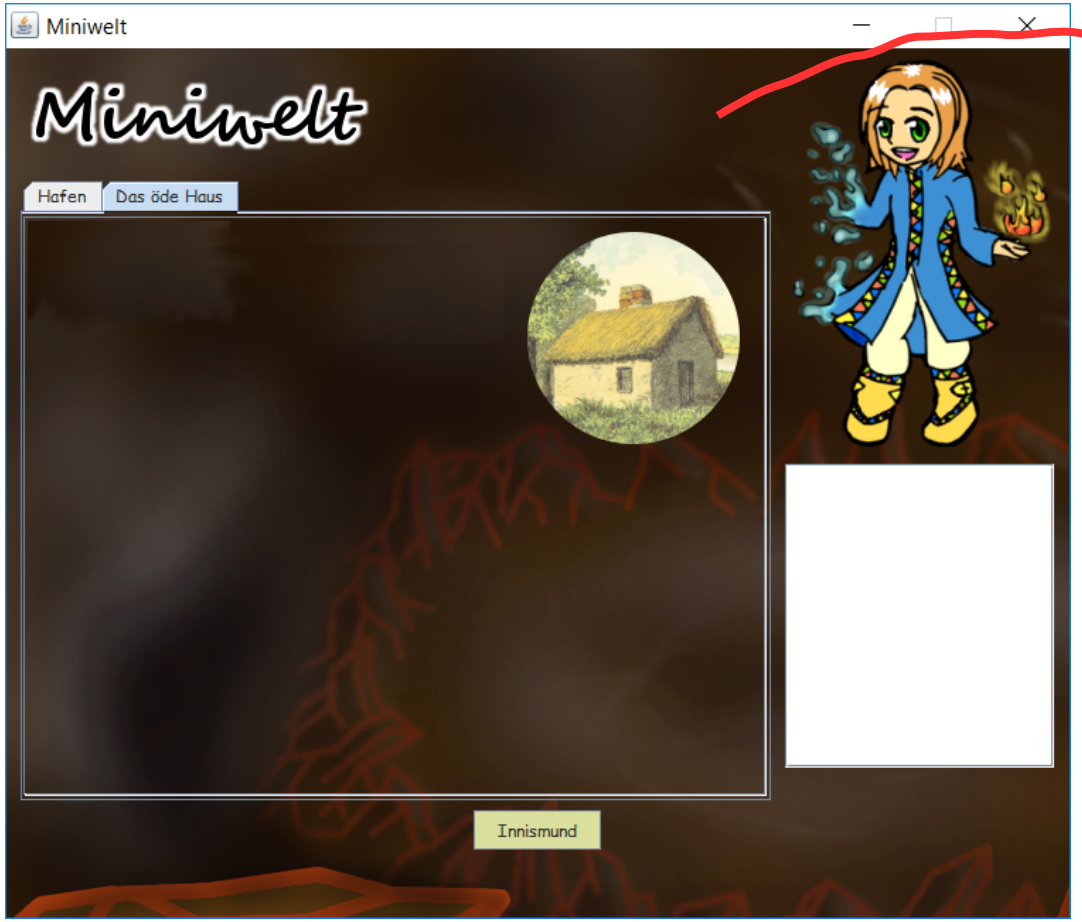
1. **Spieler betritt Ort „Hafen“.**
2. Geschichte an diesem Ort: Aufruf von createReport()
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.




```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. Geschichte reagiert...
7. ...und wird sofort danach um erneuten Report gebeten.

1. Spieler betritt Ort „Hafen“.
2. **Geschichte an diesem Ort: Aufruf von createReport()**
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.

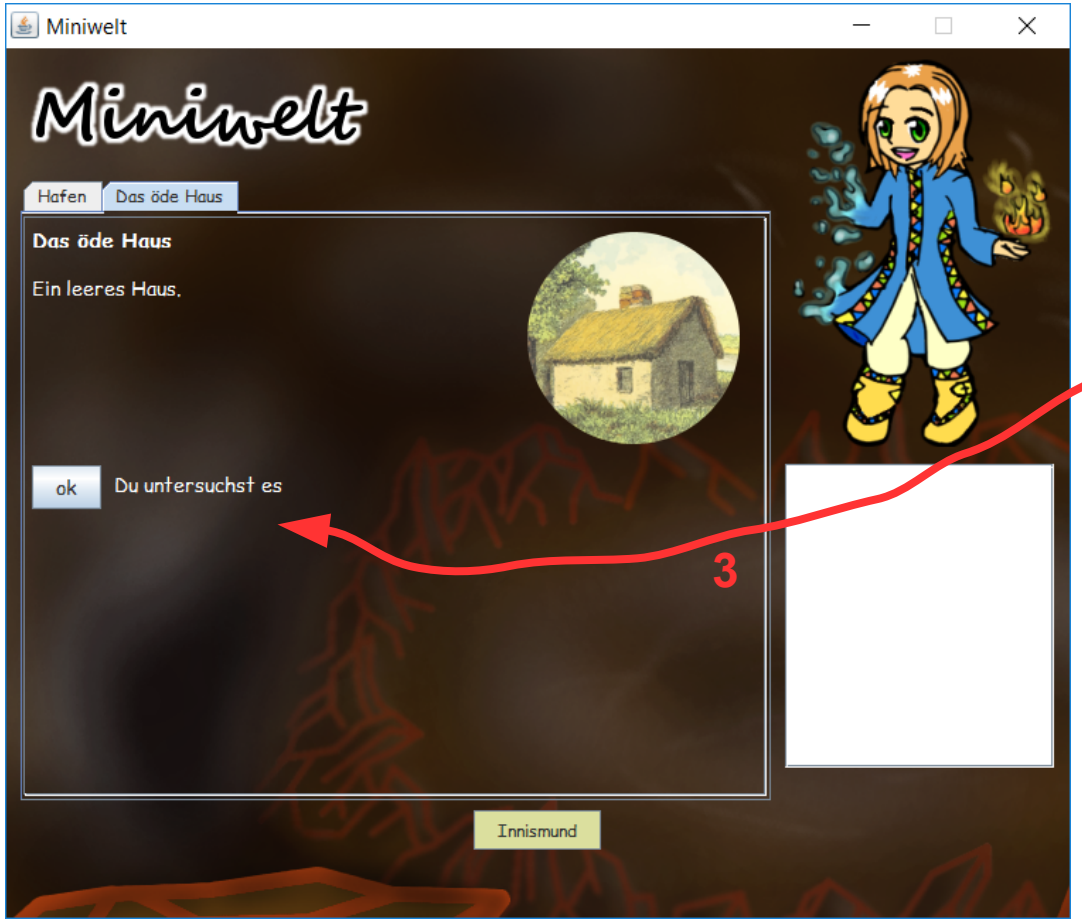


2

```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. Geschichte reagiert...
7. ...und wird sofort danach um erneuten Report gebeten.

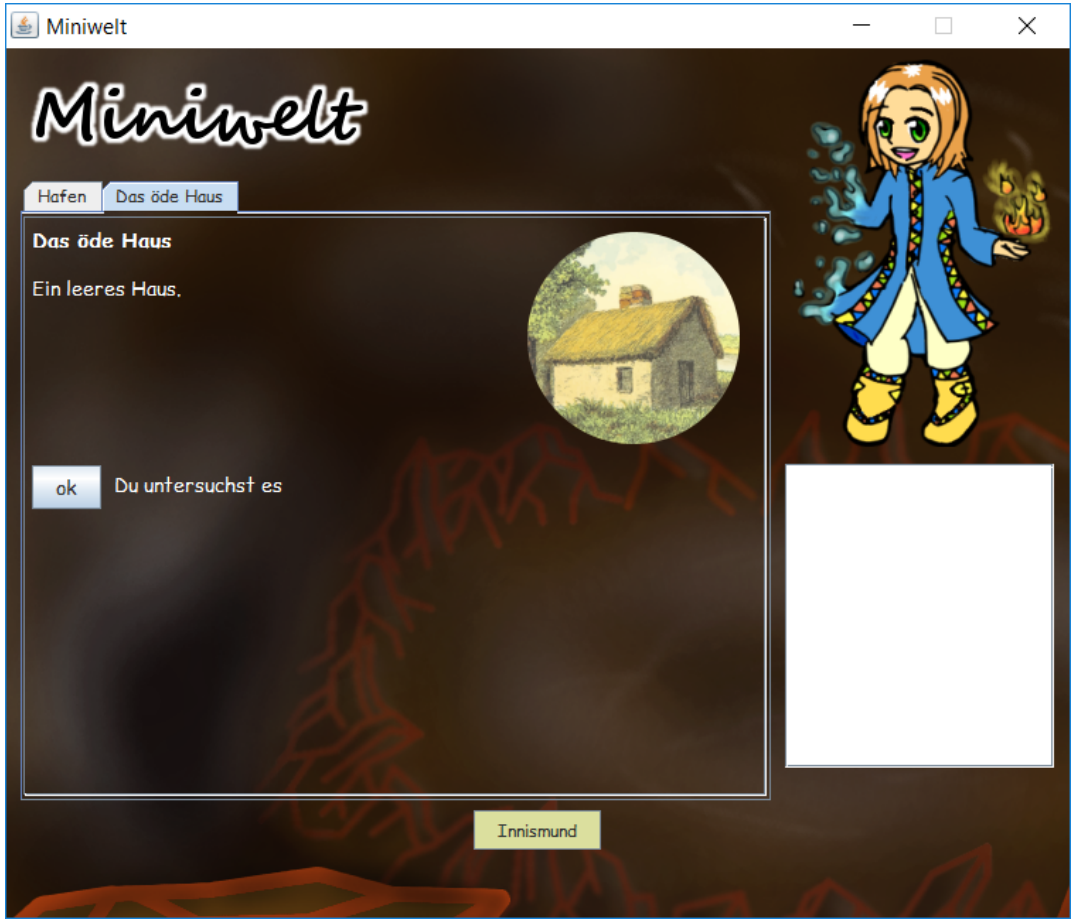
1. Spieler betritt Ort „Hafen“.
2. Geschichte an diesem Ort: Aufruf von createReport()
3. **Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.**




```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. Geschichte reagiert...
7. ...und wird sofort danach um erneuten Report gebeten.

1. Spieler betritt Ort „Hafen“.
2. Geschichte an diesem Ort: Aufruf von createReport()
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.




```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.

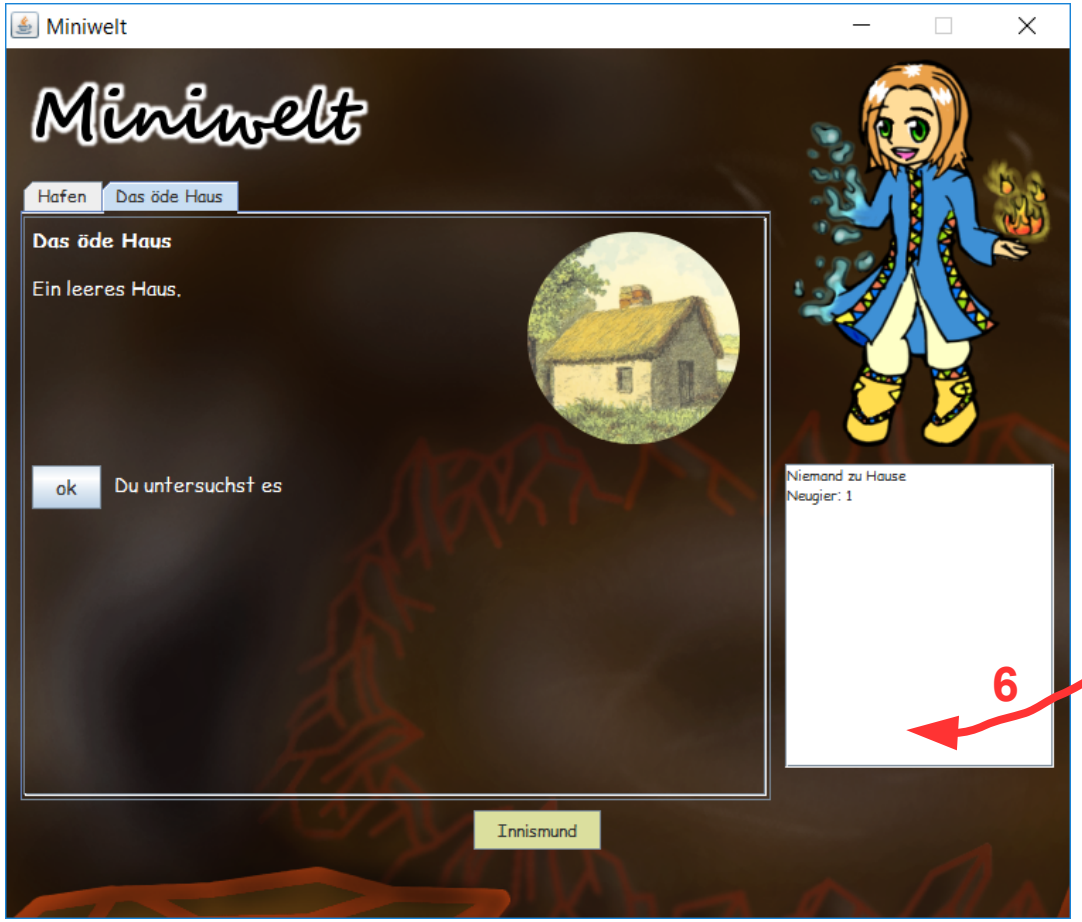
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. Geschichte reagiert...
7. ...und wird sofort danach um erneuten Report gebeten.

1. Spieler betritt Ort „Hafen“.
2. Geschichte an diesem Ort: Aufruf von createReport()
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.


```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. **Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.**
6. Geschichte reagiert...
7. ...und wird sofort danach um erneuten Report gebeten.

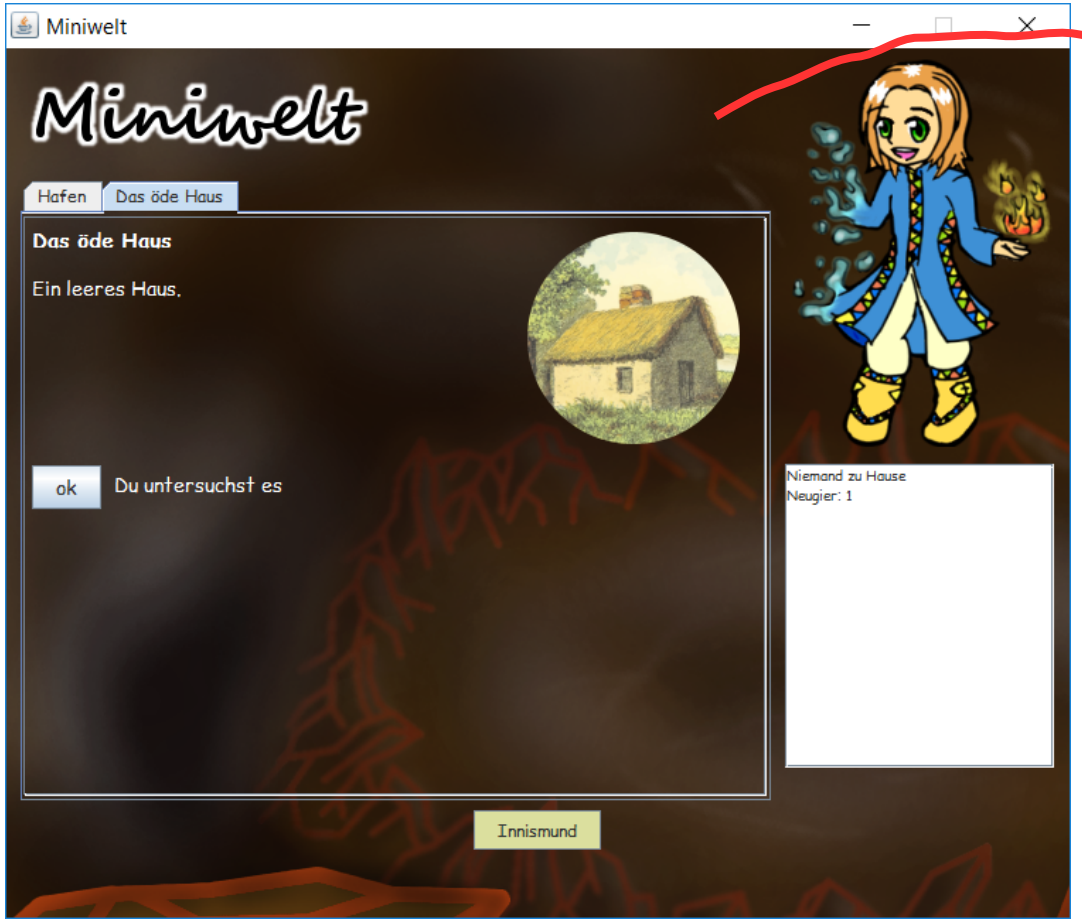
1. Spieler betritt Ort „Hafen“.
2. Geschichte an diesem Ort: Aufruf von createReport()
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.




```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
    if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. **Geschichte reagiert...**
7. ...und wird sofort danach um erneuten Report gebeten.

1. Spieler betritt Ort „Hafen“.
2. Geschichte an diesem Ort: Aufruf von createReport()
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.

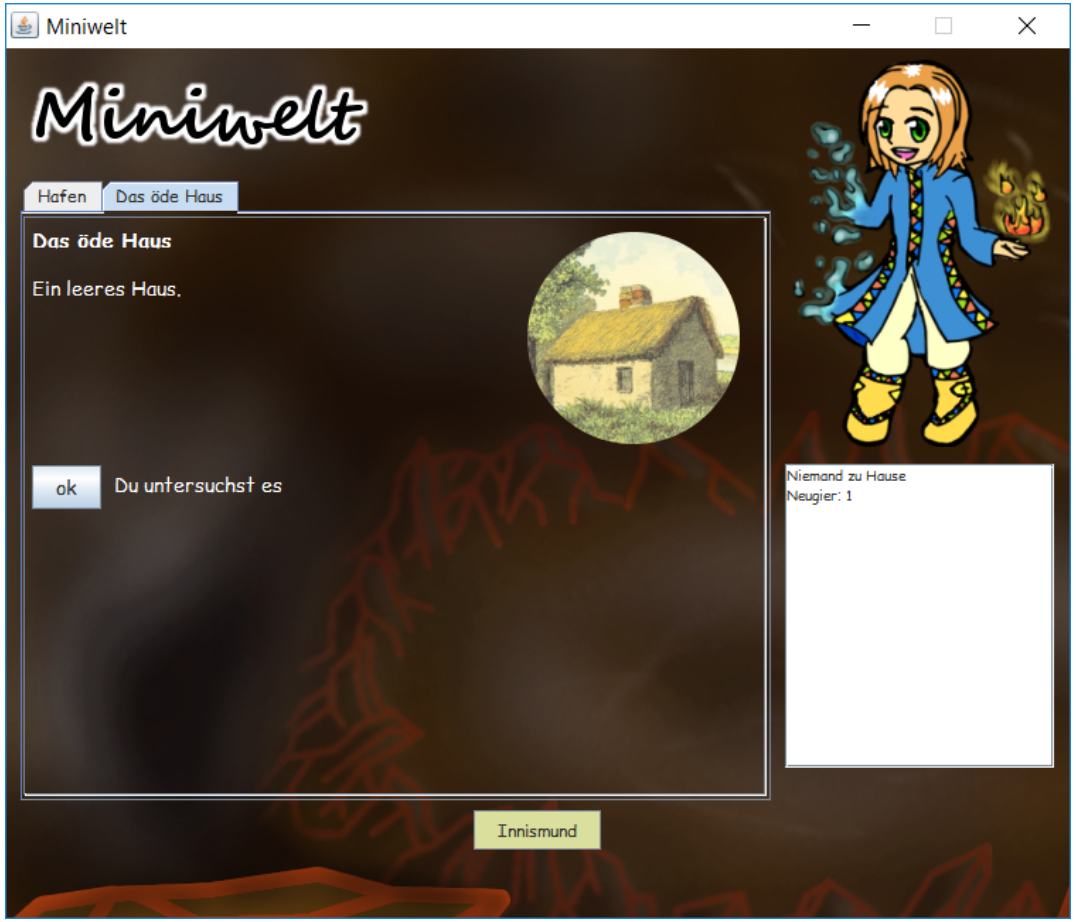


7

```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. Geschichte reagiert...
7. **...und wird sofort danach um erneuten Report gebeten.**

1. Spieler betritt Ort „Hafen“.
2. Geschichte an diesem Ort: Aufruf von createReport()
3. Liefert Report ab, der im entsprechenden Reiter dargestellt wird, mit allen Optionen.




```
class LeeresHaus3 extends SimpleStory {

 Report createReport() {
  Report r = new Report("Ein leeres Haus.");
  r.addOption("Du untersuchst es", "anschauen");
  return r;
 }

 void receiveMessage(String s) {
  if (s.equals("anschauen")) {
   sendMessage("Niemand zu Hause");
   if(get("Neugier")<10)increase("Neugier", 1);
  }
 }

}
```

4. Spieler klickt Knopf neben einer Option.
5. Methode receiveMessage wird mit der Nachricht der Option als Parameter aufgerufen.
6. Geschichte reagiert...
7. ...und wird sofort danach um erneuten Report gebeten.