Softwareentwicklungspraktikum für Fortgeschrittene

Einführungsveranstaltung

Prof. Dr. Dirk Beyer,
Matthias Dangl, Karlheinz Friedberger

Outline

I. Was wir bieten

- Produkt
- Prozess
- Technologien

II. Was wir erwarten

- Fundierte Programmierkenntnisse
- Eigenverantwortliches Arbeiten
- Zeitmanagement

III. Ablauf des Praktikums

- Zeitplan
- Benotung
- Test

Teil I. Was wir bieten

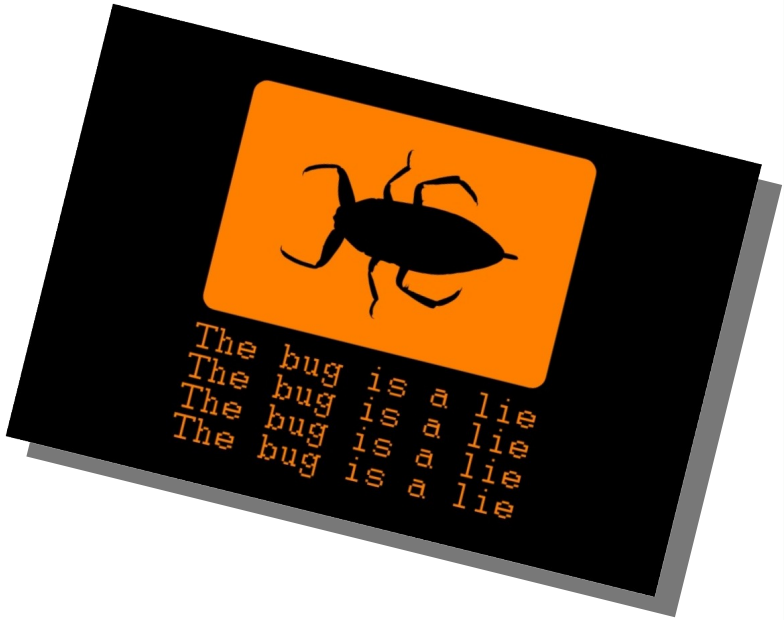
I.1 Produkt

I.2 Prozess

I.3 Technologien

Produkt

- **Ziele**
  - Modernes Produkt
  - Spaß
  - Klare Anforderungen



- Das Produkt: „The Bug Is A Lie“
  - Ein Kartenspiel, welches sich in satirischer Weise der Softwareentwicklung in der Industrie nähert
  - Eigenentwicklung des Lehrstuhls
  - Umsetzung als **Multiplayer-(„Roleplaying“)-Onlinespiel**

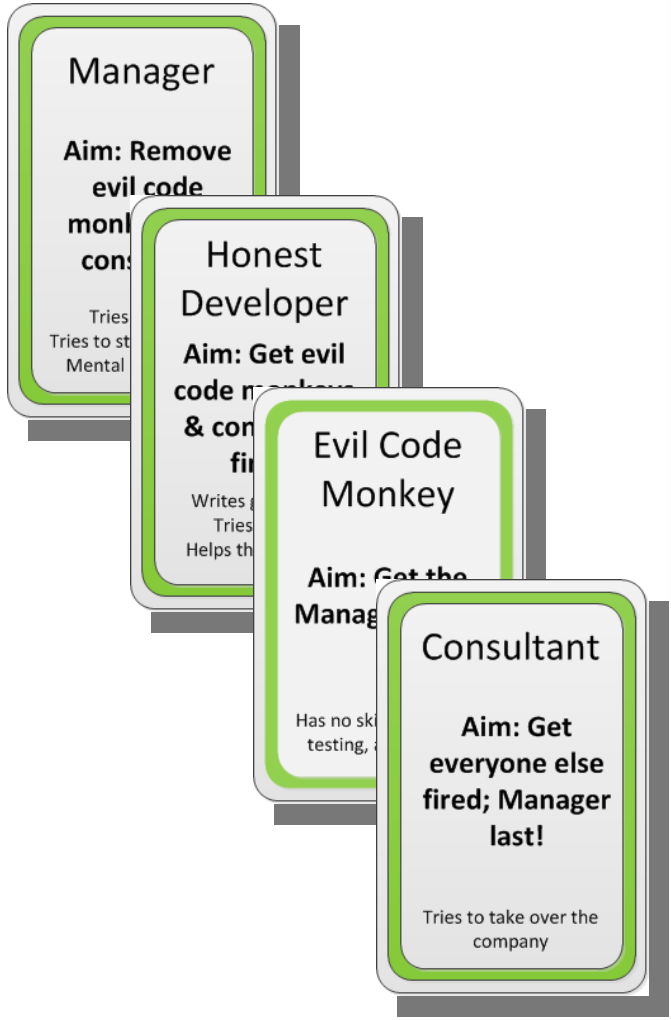
The Bug Is A Lie (I)

- Spielablauf
  - Spieler schieben sich gegenseitig **bug reports** zu, die mittels **solution** bearbeitet oder mit einer **lame excuse** abgelehnt werden müssen.

- Spieler besitzen eine gewisse Menge **mental health**; diese verringert sich bei nicht bearbeiteten bug reports. Bei **mental health == 0** wird der Spieler gefeuert.

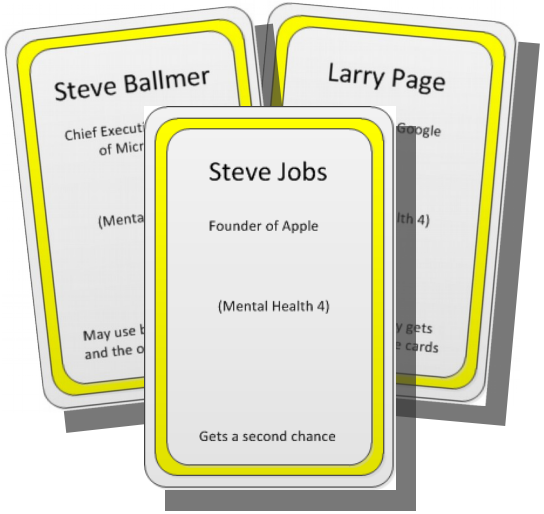
The Bug Is A Lie (II)

- Jeder der 4-7 Spieler erhält eine Rolle.
  - **Manager**
  - **Honest Developer**
  - **Evil Code Monkey**
  - **Consultant**
- Rollen (bis auf **Manager**) sind am Anfang unbekannt.
- Ziel des Spiels
  - Manager und Honest Developers: Evil Code Monkey und Consultant feuern
  - Evil Code Monkeys: Manager feuern
  - Consultant: Projekt übernehmen (=> alle anderen feuern)



The Bug Is A Lie (III)

- Zusätzliche **Persönlichkeitskarten** geben Sonderfähigkeiten.



- Verschiedene Zusatzkarten wie Prestige, Fortbildungen etc. fügen zusätzliche Aspekte hinzu.

Umsetzung

- Detaillierte Regeln werden als Teil des Entwicklungsprozesses bereitgestellt.

- **Umsetzung** erfolgt
  - als Web-basiertes Onlinespiel
    - Lobby, Spieltisch, Chat, ggf. Statistiken
  - Implementierung des Browser-Clients mittels Wicket, jQuery, HTML 5 und CSS
  - Implementierung des Servers mit Java Servlets (Anbindung an UI durch Wicket)

I.1 Produkt

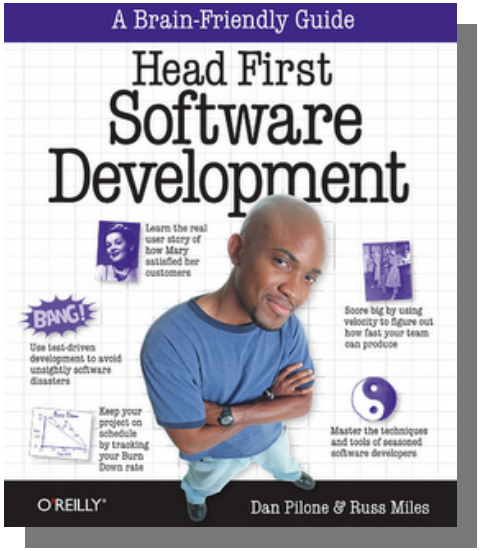
I.2 Prozess

I.3 Technologien

Prozess

- **Ziele**
  - Die Idee ist, den einen kompletten Software-Entwicklungsprozesses „in sicherer Umgebung“ durchzuführen
  - Ziel ist das Erlernen von Prozess-Techniken (Planung, Testen, Bugtracking, …)

- Der eingesetzte Prozess ist ein
  agiler Softwareentwicklungsprozess
  - Bekannte Vertreter: Scrum, XP, Kanban
  - Hier: Scrum + XP-Elemente
  - Wird in einer eigenen Vorlesung vorgestellt; Buch dazu ist jedoch (sehr!) empfohlen
  - Außerdem Einübung in einer „Lego4Scrum“ Session



Umsetzung

- Prozessablauf
  - Der Prozessablauf ist in vier **Iterationen** (2-3 Wochen) und ein **Release** (3 Monate) aufgeteilt.
  - Jede Iteration und jedes Release produziert einen **voll lauffähigen Zwischenstand** (wenn auch natürlich noch nicht mit kompletter Funktionalität versehen).

- Handling von Funktionalität
  - Umzusetzende Funktionalität wird in **User Stories** beschrieben.
  - User Stories werden **priorisiert** und **geschätzt.**

Teamwork

- **Teamwork**
  - Das Team organisiert sich im Prozess selbst
    - Volle Autorität über Planung und Umsetzung
    - Aber auch volle Verantwortlichkeit für das Produkt
  - Offenheit und Ehrlichkeit prägen die Kommunikation
- Ein Teammitglied übernimmt die Rolle des Scrum Masters
  - Scrum Master wird in jeder Iteration gewechselt
  - Kein Vollzeitjob!
- Das Team hat Product Owner (Kunde) als Ansprechpartner

I.1 Produkt

I.2 Prozess

I.3 Technologien

Technologien

- **Ziel:**
  - Einsatz **produktiver** Werkzeuge (IDEs, Git, …)
  - Einsatz **moderner** Technologien (Wicket, Mockito, …)
- Im Speziellen:
  - **Eclipse** als IDE und Build-System
  - **Technologien:**
    - Java
    - Wicket, jQuery
    - JUnit, Mockito, JaCoCo (Testing, Mocking, Coverage)
    - Selenium (System-Tests)
  - **Verwaltungstools**
    - Git, GitLab der RBG, Continuous-Integration (CI), Mylyn

Einführung in verwendete Techniken, Technologien und Tools

- Bootstrapping
  - Ein Code-Grundgerüst wird bereitgestellt; bietet Vorlage, initialen Code, und „gutes Beispiel“
  - Infrastruktur (GitLab, CI-Server und Eclipse) wird zur Verfügung gestellt.

- Einführung in Techniken, Technologien und Tools
  - Die Technologien werden in eigenen Vorlesungsstunden vorgestellt
  - Das Code-Grundgerüst wird zu Anfang gemeinsam besprochen

Teil II. Was wir erwarten

II.1 Programmierkenntnisse

II.2 Eigenverantwortliches Arbeiten

II.3 Zeitmanagement

- Fundierte Programmierkenntnisse in Java
  - Dies ist kein Java-Kurs - wir erwarten solide Java-Kenntnisse
  - Sie müssen in der Lage sein, Funktionalität umzusetzen und sich mit neuen Bibliotheken vertraut zu machen
- Begründung
  - Es geht um den Prozess der Softwareentwicklung; nicht um Coding (und schon gar nicht um Code-and-Fix)
  - Niveau der Teilnehmer soll ungefähr gleich und adäquat für die Aufgabe sein.
  - Teamarbeit sollte nicht zu stark durch Unterschiede in der Vorbildung beeinflusst werden.

II.1 Programmierkenntnisse

II.2 Eigenverantwortliches Arbeiten

II.3 Zeitmanagement

Eigenverantwortliches Arbeiten

- Wir erwarten **Interesse am Prozess und Produkt** (und die selbständige Einarbeitung in unbekannte Themen)
- Genauer:
  - Befolgung der Regeln des Prozesses (nur so lernen Sie etwas)
  - **Selbstmanagement** des Entwicklungs-Teams:
    Das Team setzt sich seine Ziele selbst und überwacht die Zieleinhaltung
  - Übernahme von **Verantwortung für das Produkt**
  - **Offenheit** und **Respekt**: Kommunikation von Problemen und Erfolgen, respektvoller Umgang mit anderen Team-Mitgliedern
  - Rolle des Scrum Masters wird von Teilnehmern übernommen

- Wichtig: die Betreuer sind nur **Kunde** (Product Owner)!

II.1 Programmierkenntnisse

II.2 Eigenverantwortliches Arbeiten

II.3 Zeitmanagement + Projektdurchführung

Zeit

- Softwareentwicklung, agile Prozesse im Besonderen, erfordert klare Richtlinien und Ehrlichkeit im Umgang mit Zeit
  - **Zeitplanung**: Ehrliche Schätzungen der Zeitaufwände
  - **Zeiteinsatz**: Nutzung nur der vorhandenen Arbeitszeit
    - Dies sind gemäß der ECTS 13 Stunden (=1,6 Tage) pro Woche
    - **Wir erwarten den Einsatz dieser 13 Stunden!**
      (aber auch nicht mehr – das ist auch eines der Ziele des Prozesses)
  - **Controlling**: Tracking der eingesetzten Zeit und Korrektur eigener Schätzungen

Projekt und Produkt

- Realistische Projektdurchführung:
  - Kein „Blame Game“
  - Bug-Reports
    - Sollen dem Prozess entsprechend behandelt werden
    - Korrekter Umgang mit Bugs wichtiger als die Anzahl der Bugs
  - Abschätzungen und Zusagen
- Entwickelte Software
  - Qualität von Design, Code, Tests
  - Deployment
  - Wartbarkeit

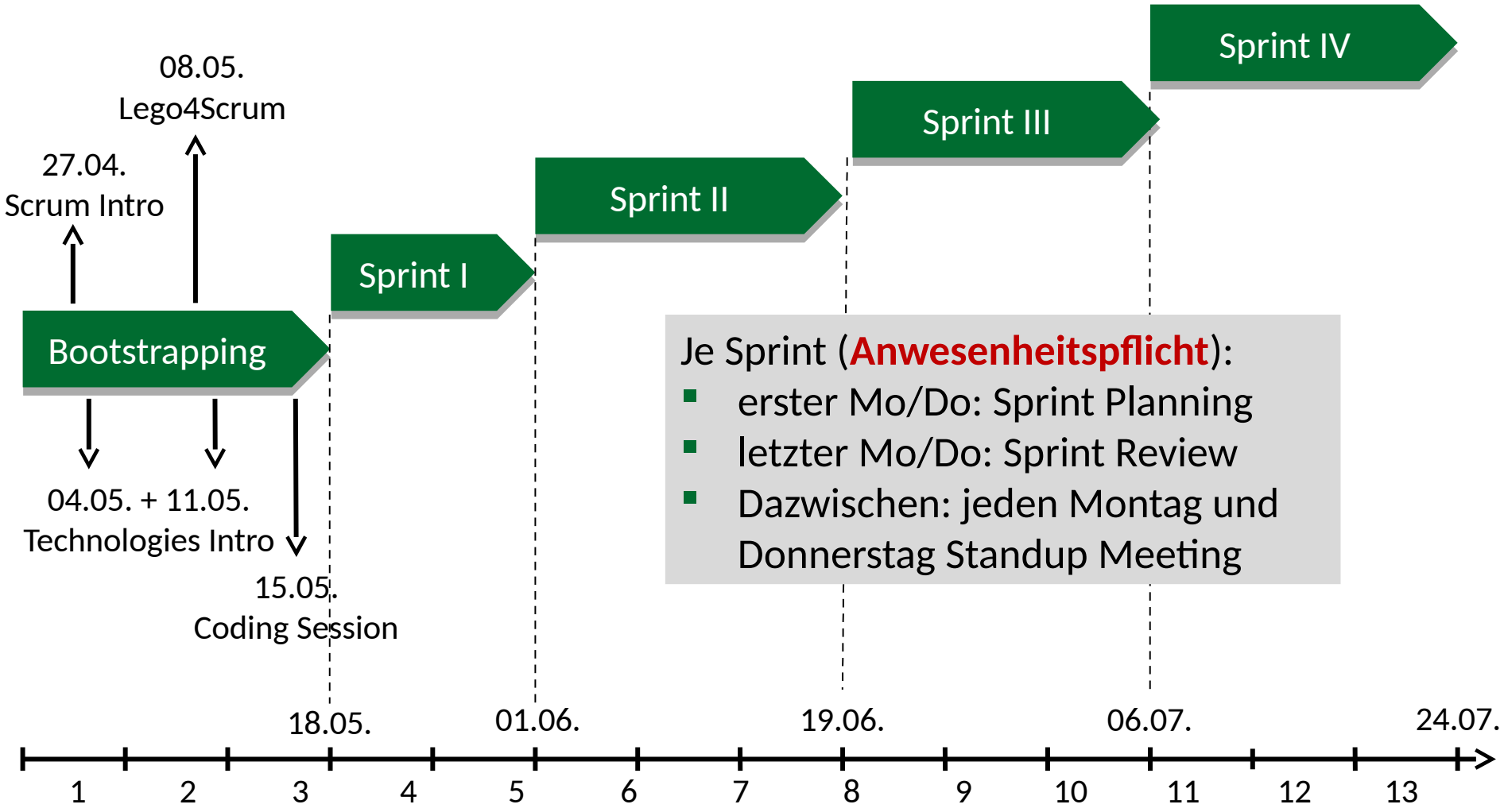
Teil III. Ablauf des Praktikums

III.1 Zeitplan

III.2 Benotung

III.3 Test

Zeitplanung



Je Sprint (**Anwesenheitspflicht**):

- erster Mo/Do: Sprint Planning
- letzter Mo/Do: Sprint Review
- Dazwischen: jeden Montag und Donnerstag Standup Meeting

- **Reguläre Termine während des Semesters**

- Voraussichtlich:
  Montags 12-16 Uhr in Raum U 139
  Donnerstags 12-14 Uhr in Raum C003 und in Raum U133

Teams und Meetings

- Zwei Teams zu jeweils 5-8 Teilnehmern
  - Alle Teams entwickeln je ein eigenes Produkt
  - Durch die Vorgaben des Produkt-Owners werden sich die Ergebnisse der Teams unterscheiden
- Wöchentliche Meetings (**Anwesenheitspflicht**)
  - Zu Beginn des Semesters:
    Einführungen in die verschiedenen Themen
  - Nach Beginn der Sprints: 2 Standup Meetings (15 Minuten)
  - …
- Abschluss: **Mo 27.07. 10-14 Uhr**

III.1 Zeitplan

III.2 Benotung

III.3 Test

Bewertungskriterien

Nicht Umfang, sondern Vorgehensweise zählt

1. **Umsetzung des Prozesses**
   - **Ehrlichkeit** und **Offenheit**
   - Zeitmanagement
   - Projektplanung
   - Berücksichtigung aller Schritte bei der Umsetzung
     (Entwurf, Test, Review, Integration, Dokumentation)
   - Vollständiger Iterationsabschluss
2. **Vorgehen in der Entwicklung**
   - Architekturentwurf und Systembeschreibung
   - Angemessene Testabdeckung
   - Dokumentation des Codes
   - Code-Qualität (Verständlichkeit und Änderbarkeit)

- Wann ist das Praktikum für Sie **nicht** die richtige Veranstaltung?
  - Falls Sie am einfachen Verdienen von ETCS-Punkten interessiert sind
  - Falls Sie einen Code-and-Fix-Ansatz der disziplinierten Software-Entwicklung vorziehen
  - Falls Sie keine Lust auf Zusammenarbeit mit anderen Menschen haben

III.1 Zeitplan

III.2 Benotung

III.3 Test

Test

- Test: 60 Minuten Java-Entwicklung am Rechner gemäß einer vorgegebenen Aufgabenstellung
- Ziele des Test
  1) Erfüllen Ihre Programmierkenntnisse unsere Anforderungen? (bezüglich Java, Nutzung von Bibliotheken, IDEs, ...)
  2) Wo stehen unsere Teilnehmer insgesamt? (Feedback für uns)
- Platzvergabe
  - Bedingung ist das erfolgreiche Bestehen des Tests
- Benachrichtigung: Donnerstag per E-Mail (UniWorX); nächstes Treffen dann Donnerstag, 27.04.

Zusammenfassung

The Last Slide

- Überlegen Sie sich,
  - … ob das Praktikum für Sie interessant ist
  - … ob Sie unsere Erwartungen erfüllen

- **15 Minuten Pause**

- Danach: Test!