JKU

JOHANNES KEPLER
UNIVERSITY LINZ

Institut fuer Wirtschatsinformatik
Software Engineering

Praktikum Software Engineering

Antonio Garmendia

Unit 1 – Project Organisation

JOHANNES KEPLER UNIVERSITY LINZ

Agenda

- **Scrum Framework**
- **Zenhub for Github**
- **Workflow**
  - Defining Issues
  - Projects, Epics and Issues
  - Reports

SCRUM

- **Framework for implementing agile principles**
  - Best practice for software development projects
- **Needs tools for successful implementation**
  - In this course, we will use Zenhub

Defining Issues

- **Do not think that the issues are just a merely "*big list of problems*"**
- **High-quality issues**
  - Well-managed
  - Triangled (Degree of urgency)
  - Labeled issues

As a Result

- Incredible insight into your code
- Track code problems
- Contributions

How to make a good issue?

- **Tell the story: “*who, what, and why*”**
- **Template Example:**

“As a <user type>, I want to <task> so that <goal>.”

How to make a good issue?

- Tell the story: “*who, what, and why*”
- Template Example:

  “As a <user type>, I want to <task> so that <goal>.”

- For instance:

  “As a <Supervisor>, I want to <Visualize an smart room> so that <I can take actions based on the real time data>.”

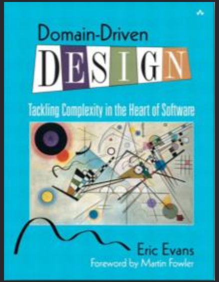
- Tutorial:

  https://help.zenhub.com/support/solutions/articles/43000074624-create-multiple-new-issue-templates-in-a-workspace

Qualities of a good issue

- Avoids jargon or mumbo jumbo. It is advisable to use “*Ubiquitous Language*”

Concept of Eric Evans


Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler

- Value to customers
- Small task that can be easily estimated in terms of time and resources required
- If it possible each issue should be independent
- Is measurable; you can test for results

Definition of Done

- **Define requirements for finishing a ticket**
- **For Issues: Acceptance Criteria**
  - Derived from user stories (cf. slide 7)
  - Best practices: https://www.productplan.com/glossary/acceptance-criteria/
  - Examples: https://agileforgrowth.com/blog/acceptance-criteria-checklist/
- **For Issues: Technical Criteria**
  - E.g. code coverage
- **For Epics: All tickets (including bugs) are done**
  - All associated tickets (including bugs) done

Sprint Planning

- **Story Points to measure effort**
- **Assign Story Points to each ticket**
  - „Planning Poker“ to decide on story points
- **Match of Story Points with Sprint Velocity**
- **Estimate Sprint Velocity**
- **Sprint Review: compare actual velocity (completed tickets) with estimate**

⚠ **Each team member must implement the same number of Story Points – Equally distributed implementation tasks!**

Projects, Epics and Issues

- Demo
- https://github.com/jku-win-se/teaching-2022.prse.project.management.example

Reports

- Roadmap
- Optional
  - Burndown Report
  - Velocity Tracking

Bibliography

- **Github Issues**
  - https://guides.github.com/features/issues/
- **Zenhub**
  - Roadmap: https://help.zenhub.com/support/solutions/articles/43000539465-an-introduction-to-zenhub-roadmaps
  - Estimating work using Story Points: https://help.zenhub.com/support/solutions/articles/43000010347-estimate-work-using-story-points

Another interesting links

- **Zenhub**
  - Zenhub free eBooks: https://www.zenhub.com/resources#ebooks
  - An Introduction to Zenhub Sprints: https://help.zenhub.com/support/solutions/articles/43000611544
  - Use Control Charts to Review Issue Cycle/Lead time: https://help.zenhub.com/support/solutions/articles/43000300345-use-control-charts-to-review-issue-cycle-lead-time