# Awesome Intro Slide

(this slide is intentionally awesome)

Google+/Meetup/etc.: modparlor@gmail.com

E-Mail: office@richdale.de

Web: www.richdale.de

Mobile/Telegram/Signal: +49 (0) 171 6 555 090

I DO *NOT* USE FACEBOOK!

This Talk

How to do

# Scrum

without killing yourself

# Right off the bat:

This is me.
There are many Scrums, but this is mine.
YMMV!

Ask 3 Scrum Masters, get 5 answers.

Do your thing. All teams are different.
All scenarios are different!

# Where I come from, where I'm at, Scrum wise

Back in the days:

18 months Senior Dev and Scrum Master with a large Game Development Company in a product team

500+ corporate crew, internal POs/customers only

(can be just as pesky)

These days (one example):

Workplace org with lightweight workplace management, kinda-sorta-Scrum in an agency online dept.

# What is Scrum? * 1/2

Scrum is a method, or variing set of methods, of formalising the organisation of teamwork in an compareatively unpredictable environment in such a way that social interaction and quick reaction is facilitated without disrupting regular workflows.

*_my_ definition, spare me the academic nitpicking

This is achieved by strictly seperating concerns of requirements analysis, goal definition, task definition, work organisation, task fulfillment and delivery.

Core element is a daily timeboxed team meeting called „Scrum", from „Scrummage" - the bunch of people huddled together pushing and shoving about in a rugby game.

# What's this „Agile" thing everybody's talking about?

Agile - or better: agility - is a new-agey foo-foo wah-wah general approach to dealing with finiky end-customers in custom software development for shiny-clicky-touchy software and web stuff.

It's really not as bad as it sounds.

Google „Agile Software Development Manifesto" and prepare to get all warm and fuzzy inside.

# Where is Agile in Scrum?

Scrum is called „an agile method" which is the wrong term. Correctly you'd call it a „method for agile software development" - it is one of the better known ones.

The method itself is NOT agile. It's actually extremely rigid, but it makes the team agile on the outside. Which is the whole point.

The more rigid and persistent you do Scrum, the more sustainable your teams agility will be. True thing.

# When *not* to use Scrum

Building Nuclear Powerplants

Executing the Lunar Apollo Missons

Building a new Airport in Berlin
(although it might be better than no method at all ... >HarHar<)

Programming clearly spec'd mission-critical ERP / Financial /
Banking / Medical / Military / Spacecraft software

# When you *should* use Scrum

Dealing with Type A websoftware clients.

They don't know what they want but they know exactly when it has to be finished and what it may cost.

When you don't really know what your up to but you know the technology you want to play with. And nobody will get hurt or die if you screw up.

# When you *should* use Scrum

Working in an environment where unforseen events are commonplace. ... Ever worked in a Theater? Fashion? Showbizz? Media Agency? ... You get the picture.

In a nutshell:

When your team needs to be agile. Duh.

# The Actors

# Team Member

It's all about the team.

No more than 9. No less than 4, Scrum Master included.

# Scrum Master

Keeps the process clean and running. Snaps at the boss/PM/PO if they disturb the team in a sprint. Is never concerned with or responsible for budget.

Is responsible for the process. Can be a team member. Tasks, complexity, sellables, features, burn-down, etc. are his daily game.

# Product Owner

The guy who utilises the final product. Or represents the customer as a PM. Deals with the outside world.

Only interested in sellables he can use or show off.

# The Meetings

# Scrum

Daily meeting. Tasks are *BRIEFLY* discussed and moved around by the member responsible. Tasks are either „waiting" „in work" „testing/review" or „finished". Team only. SM leads the meeting.

Timeboxed (15 -25 min). PO/PM may attend but must shut up.

# BAM

Backlog Assembly Meeting - When required. Tasks are collected and assigned with a (preliminary) complexity. Planing Poker sessions may take place at a BAM. All present, true PO/customer might be bored to tears though - if she's not presenting her wishlist.

PM or Tech-PM leads the meeting. Not timeboxed. Can go for a day or even days in a row. (not recommended!)

# Retrospective

Once per Sprint, after Sprint has ended. Team discusses sprint, process issues and pipeline. No PM or PO allowed. SM moderates. Timeboxed. Serious issues are escaleted.

The team stays hermetic and acts as one, the Retrospective enables this.

# Software Tools

They're shit. All of them. Don't use them.
It's that simple.

**WTF???**

It's about

# SOCIAL INTERACTION!

Get your ass away from the computer and in front of a pinboard. With other humans, in RL! (Real Life, aka „that large round room with the big lamp")

# Counter Example

At GF we built a backlog manager.
Our own. All others were crap.
(we had a crew of ~30 and a 7-digit budget)

## This is a serious fringe case!
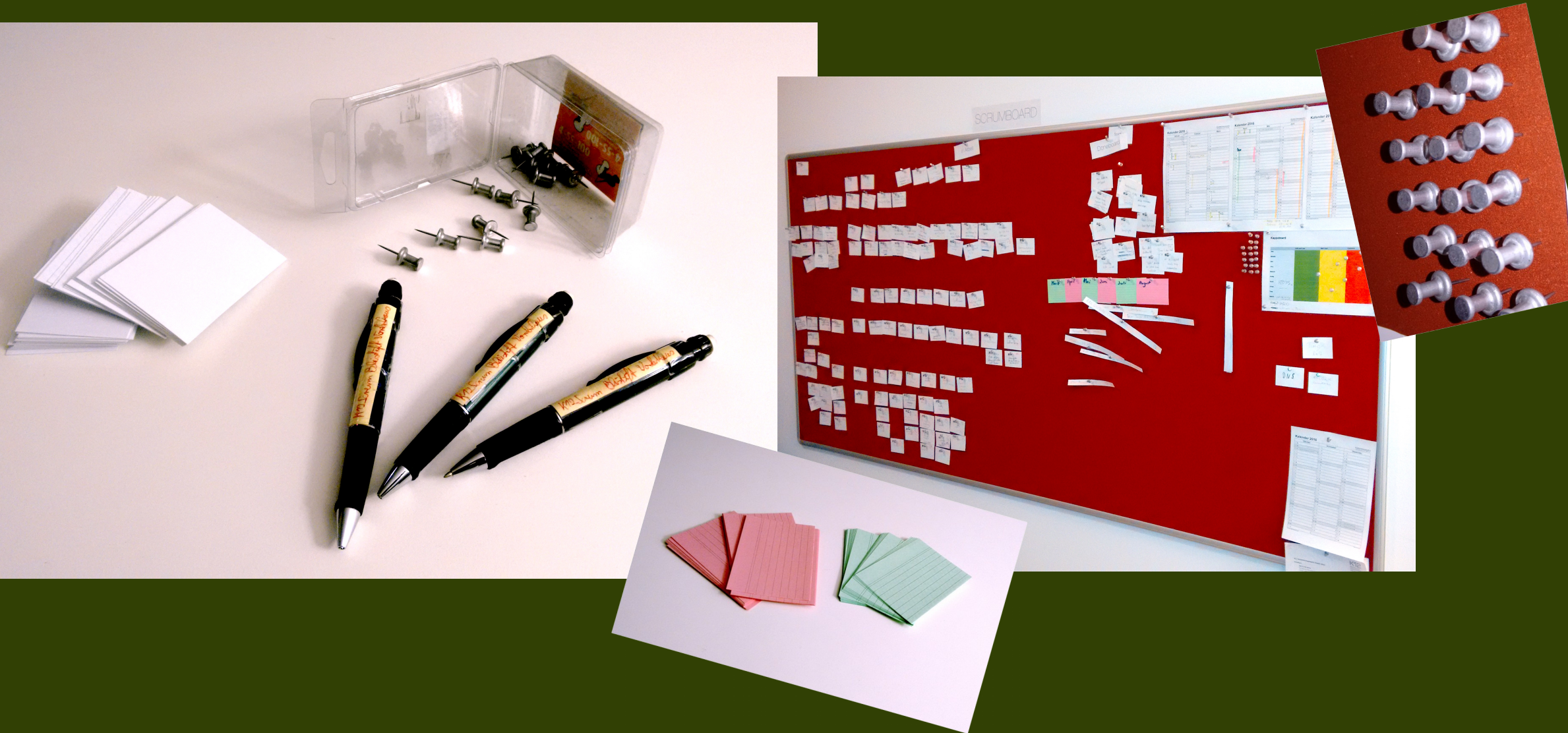
Mostly likely not applicable to other scenarios.

# Tools

Neat board. Neat pencils. Neat pins.

Neat is important! (No joke!)

Dedicated meeting space/room, pinboard, cards, pencils & time.

# Neat(!!) Tools:

# The Scrumboard

**Slots:**
- ToDo
- Work in progress
- Testing
- Finished

**Card colors:**
- Sellable: green
- Task: white
- Bug: red

# Other very neat and important stuff (Scrum and other)

- Planning Poker
- Burn Down Chart
- Project Name
  - (give your Projects unique recognizable names!!!)
- Use Cases
- Glossary
- PERT / CPM Tasking

# Recap

Don't forget: Get off your ass, away from the screen, open your mouth and don't ramble or bore.

Timebox. Always.

Scrum is about forcing everybody (including you!) to meaningful useful social interaction.

# Sprint length

Only one true answer:
One week.

# One week??!?

To short?

Possible problems:

You're not automating enough.
Your tasks are too big.
Your sellables are too big.
You don't have a working production pipeline.
You don't have a clear product strategy.

# One week ??!?

Perhaps a problem too:

You're too slow / lazy / unorganised / scared.
This is a real problem in many cases.

Rejoce. This is exactly what Scrum was built for.

# It doesn't have to be a week ...

You can Sprint twice a week if you feel like it.

I'm not joking.

# Getting Scrum into your shop (this can be tricky)

„I need it.“

„I want/would like to do/try this.“

„Let's just try this for 3 weeks and I'll writeup our conclusions.“

# Getting Scrum into your shop (handling the boss)

„Move your requests to the end/front of the week, that's easyer and I'll get more done."

„Let's have a nice pinboard with all our tasks, that way we/you can see all what we're currently doing."

# How you definitely _won't_ get Scrum into your shop

„We have to do this!"

„I've got this awesome piece of software we should use - it's nice and shiny and you can click on it ..."

„We have to do it this way ... [insert slew of complicated rules here]"

**One more thing**

Keep in mind:

# Scrum is not just for software and coding.

An Agency might have a phone call, changing the font in a PowerPoint Presentation, writing a quote or prepping a social media workshop as task on the board. You can do Scrum on a Farm too (or anywhere else it fits).

That's it.

That should get you going.

# Thanks
for listening. Questions?

# some Scrum terminology

| | |
|---|---|
| Scrum | Product Owner |
| Sprint | Scrum Master |
| Backlog | Team Member |
| Planing Poker | Task |
| Burn Down Chart | Sellable Feature |
| Complexity Rating | Bug |
| Retrospective | BAM |

# Dispersion of competence

Dispersion of competence is an element of Scrum. Everybody on the team can take on any task.

**Burn Down**

Complexity Points are tracked per Sprint and complexity burn-down progress is plotted per Sprint in a so-called „Burn-Down Chart"

After 10-20 Sprints there is a good measure of how many complexity-points the team can get done in a sprint

# Fancy Stuff 2:

Planning Poker

- Backlog has tasks that need to be estimated

- simplest task is picked and assigned a complexity of „2"

- each team-member gets a set of playing-cards with numbers on them
- some non-regular set (Fibonaci or something) : 2,3,5,8,13,20

-a task is picked by someone (usually the PO/PM or SM)

- each player picks a card with the complexity deemed appropriate

- all cards are played at the same time

- differences are discussed, if no compromise can be reached, the highest complexity is assigned to the task

# Useful Trick: *PERT / CPM Tasking

## Describe targeted state after task is completed

*Program Evaluation and Review Technique / Critical Path Method
(KAZZING! PM Buzzword Jackpot!)

# Examples

*„The application form correctly displays and highlights all cases of validation bounce as defined in Peters E-Mail from Friday.“*

*„Clicking ‚Update application‘ in the Dashboard updates the current installation to the HEAD of the release branch.“*

Team meets at the Scrumboard at 10:30

the meeting is timeboxed (15 - 25 minutes), the SM set an alarm/teatimer, everybody stands, (no leaning)

Somebody starts talking about his results from yesterday, updating the team, talking about what's up

She moves cards from „wip" to „testing" and from „testing" to „finished" as applicable, questions may arise, the team discusses details *BRIEFLY,* elaborate discussions are moved to todays tasks

She moves tasks from „waiting" to „wip" while in dialogue with the team

She ends her turn. Lather, rinse repeat. If the timer rings, the Scrum is over.

A typical Scrum

# Kanban vis-a-vis Scrum

## Scrum

## Kanban

Optimises for agility

Optimises for maximum utilisation

Is rigid

Is agile/flexible

Dispersion of competence, team spirit and a sence of responsibility is expected

Kanban is _not_ a method of agile software development!

Or at least not a very good one.