PROTOTYPING

Filip Kis
fkis@kth.se

What is a prototype?

What is a prototype?

Concrete **representation** of an interactive system/service, or *relevant* part of it

Tangible artifact

Relevance depends on what is being explored right now

Design is not linear

- We cannot say just "we do the URD and then we close that and do the SRD"
- While doing design and examining its representations, we learn more about the problem
  - In fact we often have little understanding about the problem initially
  - No amount of planning and rational thinking can replace experimentation with some design product representation
- We need ways to explore the problem without building the product
- We need to be able to go back after we understood the problem more

Prototypes and disciplines

- Architecture: scaled-down model
- Fashion: on of a kind dress
- Computer Engineering: feasibility of a technical process
- Design: express ideas and reflect on them

Why prototype?

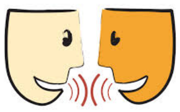
Why prototype?

- It's cheap(er)
- It's fast(er)
- It's easy
  - Can focus on the design issues rather than the technique/technology
- It allows exploration
- Reflective conversation with materials
- It's involving
- It's provocative (brings feedback)
- It's concrete (shared understanding)
  - Uncover misunderstandings early

#iprog14

Prototype roles

Creativity


Communication

Evaluation

#iprog14

Prototype roles

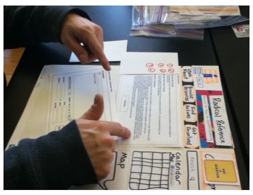
- Support creativity
  - Capture and generate ideas
  - Facilitate exploration of design space
  - Uncover relevant information about users and work practices
- Encourage communication
  - Designers, engineers, managers, developers, users, customers explore options
- Early evaluation
  - Test in various ways, through the design process

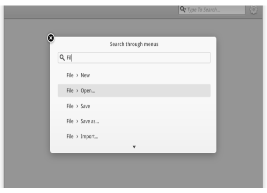
#iprog14

PROTOTYPE CLASSIFICATION

#iprog14

1. Representation


Offline



Online

#iprog14

1. Representation

- **Offline**
  - no need for a computer, or code
  - Paper sketches, storyboards, cardboard mock-ups, videos
  - Early, quick, throw-away
  - Rapid iteration and exploration
  - Prevents falling in love with first solution
  - No intermediary between idea and implementation
  - Less likely to constrain thinking due to the programming environment used
  - A wide range of people can participate
  - Increase participation and communication
- **Online (software)**
  - Computer animation, interactive video presentation, scripting, interface builder
  - Higher cost and skill
  - Later stages

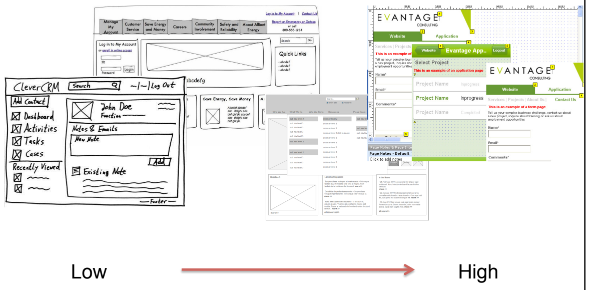
#iprog14

Prototype purpose

- The purpose varies a lot
- Depending on what is being explored right now
- Consider the purpose at each stage
- Choose the most appropriate *representation* for that purpose

#iprog14

2. Precision



#iprog14

3. Interactivity

- Interactive low-precision? Offline?
- Enactment, role-playing
- Levels of interaction
  - Fixed prototypes
  - Fixed path prototypes (limited interaction)
  - Open prototypes (large sets of interaction)
- Illustrating possible interactions is different from interactivity!

#iprog14

4. Evolution

- Rapid
  - Cheap, easy, exploring
  - Online of offline
- Iterative
  - Iterate to vary a theme
  - Iterate to increase precision
- Evolutionary
  - Iterative prototypes that evolve into the final system (or part of it)
  - Extreme programming, agile methodologies

#iprog14

Prototypes in the design process

- User-centred design
  - Users see the system before it is built
  - Multiple design-implement-test loops
- Contextualize
  - Imagine the real system in the real setting
  - Vs abstract ideas, models
- Participatory design
  - Users involved at all stages, partners, learning process
  - Sometimes data from prototyping feeds into design research

#iprog14

Prototypes in the design process (cont'd)

- Explore design space
  - Expanding (generating ideas)
    - Brainstorming (write down ideas)
    - Video brainstorming (act ideas, record)
  - Contracting (selecting alternatives)
    - Focusing on an idea, e.g. Video prototype

#iprog14

Prototyping strategies

- Horizontal prototypes
  - Cover an entire layer of the design, iterative, increasing precision
- Vertical prototypes
  - Assess the feasibility of a feature down to the lower system layers
- Task-oriented prototypes
- Scenario-based prototypes

#iprog14

ONLINE PROTOTYPING

#iprog14

When to do online prototypes?

- Beaudoin-Lafon and Mackay
  - Programmers often argue in favor of software prototypes, even at the earliest stages of design. Because they are familiar with programming languages, programmers believe it will be faster and more useful to write code rather than “waste time” creating paper prototypes

#iprog14

When to do online prototypes?

- In 20 years of prototyping, we have yet to find a situation where this is true

#iprog14

Programming for a prototype

- GUI programming is expensive, 50-80% of the total project cost
- It is more acceptable to have longer prototyping cycles in late project stages,
  - when the design space is more narrow/constrained
- But in early stages make sure that
  - Your language won’t constrain you time- or feature-wise
  - You have all needed resources (data, libraries)
- Interactive prototypes may be more suggestive to users, during evaluation
- Interactive prototypes can be played with to discover their functionality by exploration

#iprog14

Precision

- Online prototypes can be high or low precision
- Precision is the tension between
  - what the prototype states (relevant details), for evaluation
  - what the prototype leaves open (irrelevant details), for future discussion and exploration
- In early stages, it is important for the prototype to look sketchy and unfinished (Bill Buxton)
  - Many tools provide standard widgets, which look finished
- A detailed representation needs not be precise
  - Some parts may be left out or presented intentionally in a superficial way, to denote that they are open for future discussion
  - Sometimes the interaction is there, but not the functionality

#iprog14

Interactivity

- How interactive it *feels* to user
- Interactivity and precision are also orthogonal
- *Fixed* prototypes: Non-interactive
  - Yet they may *describe* interaction well
  - Video clips, pre-computed animations
  - Usually associated with a *use scenario*
- *Fixed-path* prototypes: Limited interaction
- *Open* prototypes: large sets of interactions
  - But usually cover part of the system
- *Wizard of Oz*: interactivity with simulated functionality
- Alternatively: *stub* functionality
  - Dummy implementation of a complex module

#iprog14

Evolution

- *Rapid*: for a specific purpose, thrown away
- *Iterative*: refine aspects, or explore more alternatives
  - Even an offline prototype can be iterated!
- *Evolutionary* into the final system
  - Inherent tension between exploring design space and evolution to the final system
    - Maybe combine rapid/iterative -> evolutionary
  - *Architecture* of a prototype and of the final system are often very different
  - Extreme programming (e.g. Refactoring)

#iprog14

Rapid online prototypes

- Higher precision than offline
- Some dynamic interactions are difficult to visualize offline, but easier to animate
- *Non-interactive simulations*
  - A movie of some kind.
  - Macromedia Director, or Flash
    - Scenes can be paper or computer-drawn, screenshots
  - But also PowerPoint, drawing programs (Illustrator, Photoshop),
    - Use of layers is useful to describe different phases
  - *Manual simulation*: hide and show layers, change slides

#iprog14

Rapid online prototypes

- Interactive simulations
  - Typically result in fixed-path interactive prototypes
  - Photoshop layers can also be used for interaction
  - Hypercard, the card metaphor
  - Director/Flash: behaviors attached to symbols
  - PowerPoint: Action Settings/Hyperlink
  - HTML hyperlinks
  - Scripting languages: can be cryptic but rapid (e.g. Tcl/Tk. Not strongly typed, ignore non-fatal errors
    - button.dialogbox.ok -text OK -command {destroy.dialogbox}

Click Me to go to slide 8

#iprog14

Iterative prototypes

- Even shipped product versions can be regarded as iterations
- Software tools
  - Graphical libraries and windows systems
  - User interface toolkits
    - Widget: presentation, behavior, API (callback, listener)
    - E.g. Java AWT
  - User Interface Builders
    - Placing and laying out widgets usually leads to boring programming, interface builders help
    - Editor, Run-time generator, functional core
    - Many support changing the interface while it runs (e.g. Visual Basic)
    - And compile it in less resource-intensive code later

#iprog14

Iterative prototypes (cont'd)

- Software environments
  - Application frameworks to generate specific kinds of applications (e.g. Unidraw)
  - Or just more support for windowing applications (Java Swing, etc)
  - Model-based tools (our research ☺)
  - User interface development environments
    - Silk: sketch first (as on paper), it recognizes gestures and animations

#iprog14

Evolutionary Prototypes

- A special case of iterative prototypes
- Extreme programming
- Must carefully consider software architectures
- OOP Design patterns
  - Inspired from architecture, like UI design patterns
  - Model-View Controller (MVC)
  - Presentation-abstraction controller (PAC)
  - But many other patterns are inspired from GUI programming

#iprog14

Conclusions

- Prototype with whatever language/environment allows you and your team to be fast and focus on the design, not implementation details
  - Exception: vertical prototypes where you investigate technology limitations and their UI addressing
- Consider how the design ideas will be communicated to a prototype developer who is not a design team member. Offline prototype?
- Evolutionary prototypes may seem attractive but are often not good to start with

#iprog14

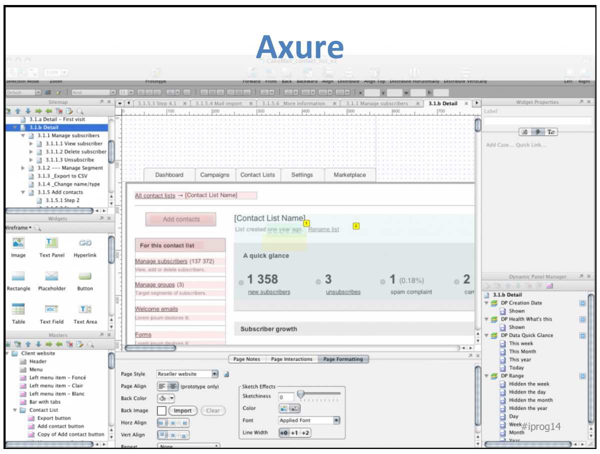
Literature

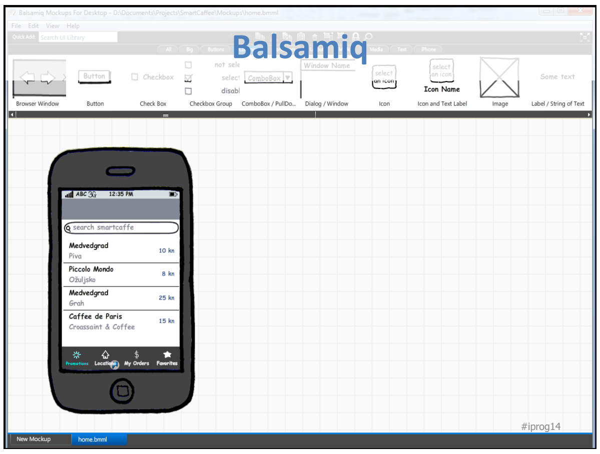
- Beadoin-Lafon and Mackay: **Prototyping tools and techniques**
- Westerlund, Bo: Design Space Exploration
- Lindquist, Sinna: Perspectives on Cooperative Design

#iprog14

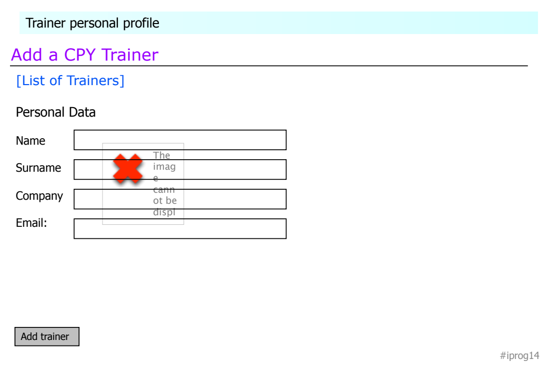
PROTOTYPE TOOLS

#iprog14





PowerPoint



Trainer personal profile

Add a CPY Trainer

[List of Trainers]

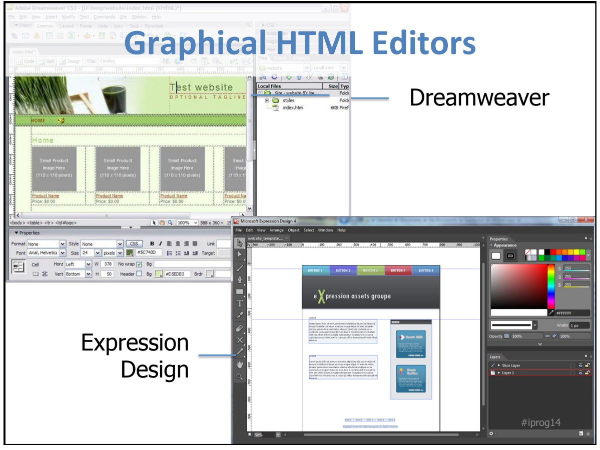
Personal Data

Name	
Surname	
Company	
Email:	

Add trainer

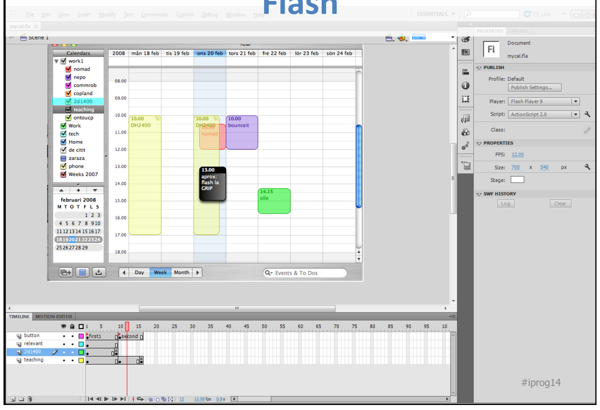
#iprog14

Graphical HTML Editors



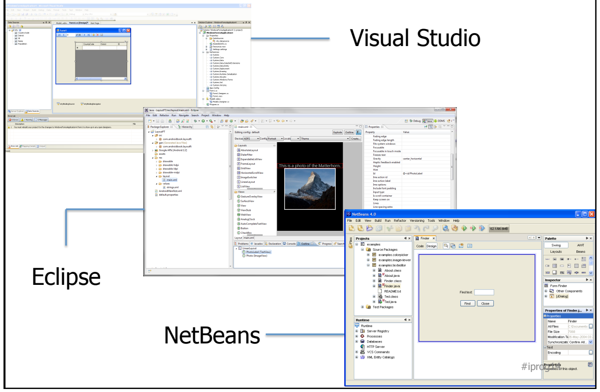
#iprog14

Flash



#iprog14

GUI Builders



#iprog14

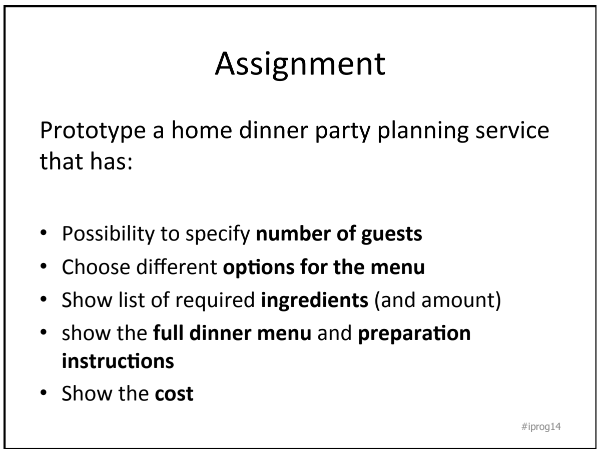
LAB 1: PROTOTYPING

#iprog14

Goals

- Practice prototyping (offline & online)
- Fast prototype iteration
- Prototype evaluation
- Choosing among alternatives
- Feedback from users and other designers

#iprog14

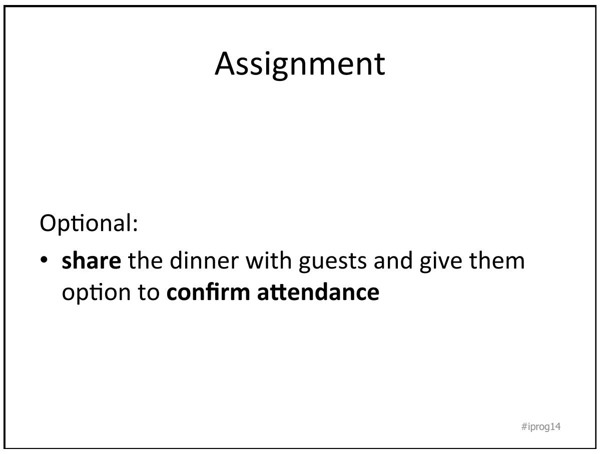


Assignment

Prototype a home dinner party planning service that has:

- Possibility to specify **number of guests**
- Choose different **options for the menu**
- Show list of required **ingredients** (and amount)
- show the **full dinner menu** and **preparation instructions**
- Show the **cost**

#iprog14

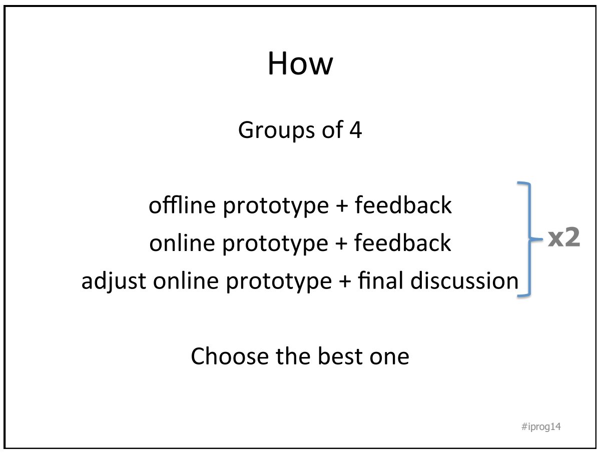


Assignment

Optional:

- **share** the dinner with guests and give them option to **confirm attendance**

#iprog14



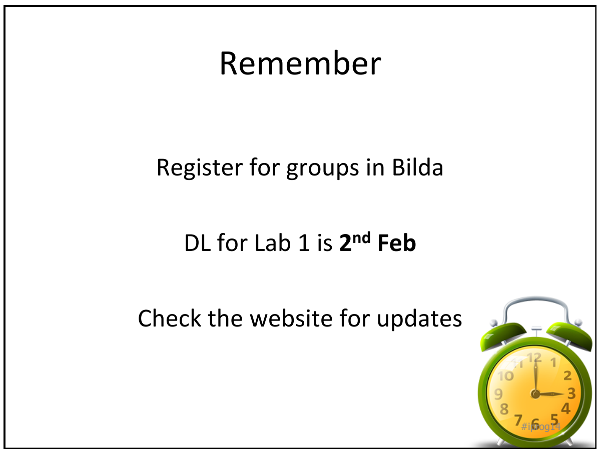
How

Groups of 4

offline prototype + feedback
online prototype + feedback
adjust online prototype + final discussion } **x2**

Choose the best one

#iprog14



Remember

Register for groups in Bilda

DL for Lab 1 is **2nd Feb**

Check the website for updates

#iprog14