Praktikum Entwicklung von Mediensystemen mit iOS

SS 2011

Michael Rohs
michael.rohs@ifi.lmu.de
MHCI Lab, LMU München
Milestones

• 26.5.
  – Project definition, brainstorming, main functions, persona
• 9.6. (week 1)
  – Identify user needs (interview or observation)
  – Storyboarding, low fidelity paper prototyping
• 16.6. (weeks 2,3)
  – Test paper prototype with users
  – Start of software prototype development
• 30.6. (week 4)
  – Heuristic evaluation of software prototype
• 7.7. (weeks 5,6)
  – Think-aloud user study on software prototype
• 21.7. (week 7)
  – Completion of software prototype, preparation of presentation
• 28.7.
  – Presentation of project results
Tasks

• Present milestone results at meetings
• Meet with your group regularly
• 9.6.
  – Present project idea, present persona, narrow down functionality
• 16.6.
  – Present interview results, storyboard, first paper prototype
• 30.6.
  – Present paper prototype test results (and plan for revision)
• 7.7.
  – Present results of heuristic evaluation (and plan for revision)
• 21.7. **today**
  – Present results of think-aloud user study (and plan for revision)
• 28.7.
  – Present complete project
EVALUATION
DIA Cycle: When to evaluate?

Design

Analyze
Evaluate with or without users

Implement
Think Aloud

• As Silent Observation, but user is asked to say aloud
  – What he thinks is happening (state)
  – What he is trying to achieve (goals)
  – Why he is doing something specific (actions)

• Most common method in industry

+ Good to get some insight into user’s thinking, but:
  – Talking is hard while focusing on a task
  – Feels weird for most users to talk aloud
  – Conscious talking can change behavior

Source: Saul Greenberg
MEMORY MANAGEMENT & INSTRUMENTS
Reference Counting

• Object reference life cycle:
  
  myobject = [[MyClass alloc] init];       // reference count = 1 after alloc
  [myobject retain];          // increment reference count (retainCount == 2)
  [myobject release];         // decrement reference count (retainCount == 1)
  [myobject release];         // decrement reference count (retainCount == 0)
  // at this point myobject is no longer valid, memory has been reclaimed
  [myobject someMethod];     // error: this will crash!

• Can inspect current reference count:
  
  NSLog(@"retainCount = \%d", [textField retainCount]);

• Can autorelease (system releases at some point in future)
  
  [myobject autorelease];
  Used when returning objects from methods.
Rules

- Memory rule: You are responsible for objects you allocate or copy (i.e. “allocate” or “copy” is some part of the name)!

- Not responsible:
  ```
  NSData *data = [NSData dataWithContentsOfFile:@"file.dat"]; 
  ```

- Responsible:
  ```
  NSData *data = [[[NSData alloc] initWithContentsOfFile:@"file.dat"];
  ```

- Responsible:
  ```
  NSData *data2 = [data copy];
  ```

- Never release objects you are not responsible for!
Objective C - Class

In .h file:

#import <Foundation/Foundation.h>

@interface Employee : NSObject
{
    // Instance vars here
    NSString *name;
    int salary;
    int bonus;
}

// methods outside curly brackets
- (void)setSalary:(int)cash withBonus:(int)extra
@end
Objective C Properties

• .h file:
  ```
  @interface MyDetailViewController : UIViewController {
    NSString *labelText;
  }
  @property (nonatomic, retain) NSString *labelText;
  @end
  ```

• .m file:
  ```
  @synthesize labelText;
  -(void)someMethod {
    self.labelXText = @”hello”;
  }
  ```

creates accessor methods: setLabelText (retains/releases) and getLabelText.

dot-syntax means: use property’s setLabelText accessor method, will retain the object

equivalent to [self setLabelText:@”hello”];
Implicit Setter/Getter Accessor Methods

- .h file: @property (nonatomic, retain) NSString *labelText;
- .m file: @synthesize labelText;
- Automatic creation of accessor methods:
  - (void) setText:(NSString *)newLabelText {
    [labelText release];
    labelText = newLabelText;
    [labelText retain];
  }
  - (NSString*) getText {
    return labelText;
  }
- Properties are accessible from other classes, data members only if declared @public
Property Attributes

- Writability: readwrite (default), readonly
- Setter semantics: assign, retain, copy
- Atomicity: atomic (default), nonatomic

- “readonly” means only a getter, but no setter accessor method is generated by @synthesize
Analyzing Code

- Xcode static analysis for simple problems
Profiling Code

- Analyzing runtime behavior
Profiling Code
Profiling Code
Best Method Avoiding Memory Leaks

• Program carefully, think hard
• Follow the memory management rules

• Ugly truth:
  Some leaks are in the frameworks as well!
Presentation Structure

• Target audience of presentation: investors
  – Imagine getting funding for a startup company

• Suggested presentation outline (7 minutes per group)
  – Group and product introduction (1 min)
  – Target user group (1 min)
  – Important features (1 min)
  – Role-play of usage scenario, presentation of interaction techniques (3 min)
  – Design process, design principles, challenges (1 min)