Higher Order Messaging

Torsten Kammer • http://ferroequinologist.de/

Was ist das?

- Begriff tauchte erstmals 2005 auf, Idee selbst ist älter
- http://www.metaobject.com/papers/Higher_Order_Messaging_OOPSLA_2005.pdf
- Methoden, die Methodenaufrufe als Parameter nehmen
- `[[array makeObjectsPerform] operationWithParameter:a b:b c:c]`

Beispiele

- Distributed Objects
  - Nachrichten werden über Netzwerk gesendet
- NSUndoManager
  - `[[undoManager prepareWithInvocationTarget:target] setState:oldState];`
  - Nachricht wird gespeichert, beim Undo gesendet

Eigene Anwendung

- Methode über Array iterieren
  - `[[array makeObjectsPerform] message:parameter];`
- Methode woanders hin schicken
  - `[[object performOnMainThread] someOperation];`
- Parameter austauschen
  - `[[Class allocEach] initWithObject:[array each]];`

Implementation

- Es gibt fertige Frameworks, z.B. Liste auf

  http://www.cocoadev.com/index.pl?HigherOrderMessaging

- Viele denkbare Anwendungen sind nicht in Frameworks
- Diese muss man selber implementieren

Grundlagen

- HOM-Methode wird aufgerufen (z.B. performAfterDelay:)
  - Speichert Daten irgendwo
- Parametermethode wird aufgerufen
  - Kann nicht normal verarbeitet werden
  - Wird zu NSInvocation
  - Wird zusammen mit HOM-Methode verarbeitet

Forwarding

```
- (NSMethodSignature *)methodSignatureForSelector:(SEL)sel
{
  return [target methodSignatureForSelector:sel];
}
```

```
- (void)forwardInvocation:(NSInvocation *)anInvocation
{
    ...
    [anInvocation setArgument:... atIndex:...];
    ...
    [anInvocation invokeWithTarget:...];
    ...
    [anInvocation getReturnValue:...];
    [anInvocation setReturnValue:...];
}
```

Trampolines

- Zwischen zwei Methodenaufrufen vergeht Zeit
  - Ersten Methode und ihre Parameter müssen noch bekannt sein
- Kategorie kann dies nicht zwischenspeichern.
- Klasse kann forwardMessage: schon anders verwenden
- Lösung: Trampolines

Ablauf

```
[[object performAfterDelay:0.0] doSomething]
```

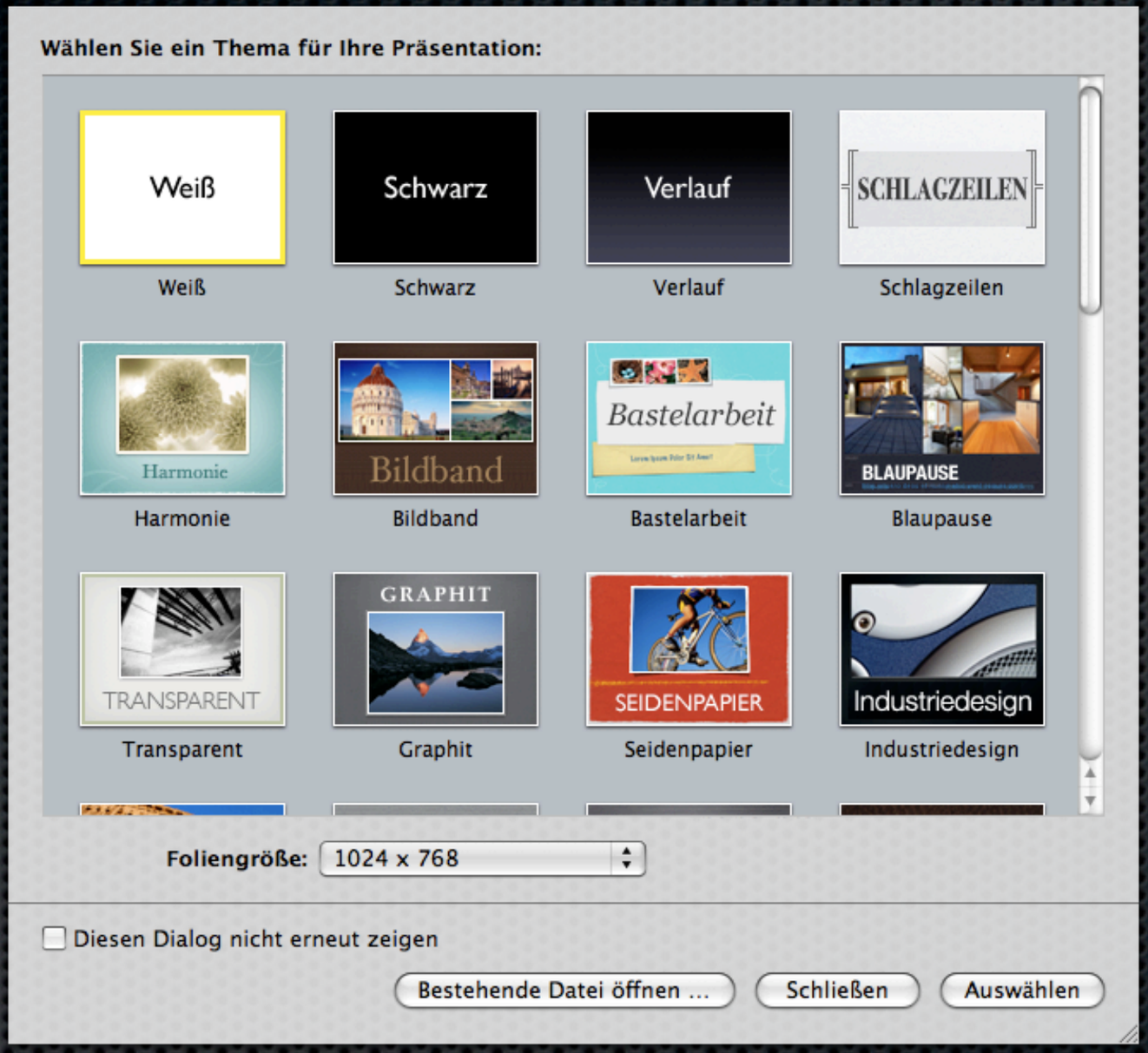
object

tram-poline

```
[object performAfterDelay:0.0]
[... doSomething]
```

Fehler

```
[... messageSignatureForSelector:]
[... forwardInvocation:]
[... performAfterDelay: withInvocation:]
```



Beispiel #1

Sheet bei neuem NSDocument

Das Problem

- Geht nicht in `windowControllerDidLoadNib:`, Fenster noch nicht auf dem Bildschirm
- Aber direkt danach
- `beginSheet:modalForWindow:modalDelegate:didEndSelector:contextInfo:` hat viele Parameter
- `performSelector:withObject:afterDelay:` überträgt nur einen
- Brauchen normalerweise Hilfsmethode

Lösung mit HOM

- Normale Lösung braucht zusätzliche Methoden, umständlich
- Statt dessen (wie eben als Beispiel):
- `[[[NSApplication sharedApplication] performAfterDelay:0.0] beginSheet: …`

Kompliziertes Beispiel

Iterieren über viele Arrays

- Beispiel: Zwei Arrays mit Strings.
  - a = @“foo“, @“bar“, @“baz“
  - b = @“a“, @“b“, @“c“
- Wollen an jedes Element aus a das entsprechende aus b hängen.

```
[[a doEach] stringByAppendingString:[b each]]
```

- Braucht zusätzliche Klasse als „each“-Markierung
- Durchsucht Argumenten-Liste (und Ziel) für diese each-Markierung, entnimmt daraus NSEnumerator
- Bis wahlweise alle oder einer der Enumeratoren nil liefert:
  - Parameter auf neues Objekt setzen
  - Methode aufrufen
  - Ergebnis in NSMutableArray speichern

Mögliche Probleme

- Rückgabewert muss stimmen
  - Kann nicht NSArray liefern wenn Parametermethode normalerweise float liefert
- Falls doch gewünscht:
  - Andere Methodensignatur liefern
  - Compiler austricksen

Weitere Möglichkeiten

- performOnMainThread, performOnNewThread
- Mehr Array-Methoden - z.B. Kreuzprodukt
- Zwei Parametermethoden mit zwei Trampolines
  - `[[[a doEach2] dataUsingEncoding:…] writeToFile:[b each]]`

Performance

- Grauenhaft
- forwardMessage: und NSInvocation ist sehr, sehr viel langsamer als ein normaler Methodenaufruf.
- Im Normalfall macht das keinen Unterschied
- Nichts für schnelle innere Schleifen