Freie Universität Berlin

Title As It Is In the Proceedings
Include Only If Paper Has a Subtitle

F. Author S. Another
Freie Universität Berlin

Conference on Fabulous Presentations, 2003

Outline

Rail-Editor: Übersicht

Übersicht:

- Softwaretechnik
- Technische Aspekte
- Live-Demo

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email
- direkte Reaktionen auf Emails (an alle geschickt)

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email
- direkte Reaktionen auf Emails (an alle geschickt)
- zuerst zwei, nach dem ersten Milestone drei Mitglieder
  - bessere Arbeitsverteilung

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email
- direkte Reaktionen auf Emails (an alle geschickt)
- zuerst zwei, nach dem ersten Milestone drei Mitglieder
  - bessere Arbeitsverteilung
- Teilnahme an Daily Scrums (montags und mittwochs)

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email
- direkte Reaktionen auf Emails (an alle geschickt)
- zuerst zwei, nach dem ersten Milestone drei Mitglieder
  - bessere Arbeitsverteilung
- Teilnahme an Daily Scrums (montags und mittwochs)
- zusätzliche Team-Meetings außer donnerstags
  - produktives Arbeiten durch Pair-Programming

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email
- direkte Reaktionen auf Emails (an alle geschickt)
- zuerst zwei, nach dem ersten Milestone drei Mitglieder
  - bessere Arbeitsverteilung
- Teilnahme an Daily Scrums (montags und mittwochs)
- zusätzliche Team-Meetings außer donnerstags
  - produktives Arbeiten durch Pair-Programming
- gute Kommunikation innerhalb des Teams

Rail-Editor: Softwaretechnik

- kleines Team → Kommunikation per Email
- direkte Reaktionen auf Emails (an alle geschickt)
- zuerst zwei, nach dem ersten Milestone drei Mitglieder
  - bessere Arbeitsverteilung
- Teilnahme an Daily Scrums (montags und mittwochs)
- zusätzliche Team-Meetings außer donnerstags
  - produktives Arbeiten durch Pair-Programming
- gute Kommunikation innerhalb des Teams
- anfangs spärliche Kommunikation mit anderen Teams

Rail-Editor: Technische Aspekte I

- *QT* für Programmierung der grafischen Benutzeroberfläche

Rail-Editor: Technische Aspekte I

- *QT* für Programmierung der grafischen Benutzeroberfläche
- Signal- und Slottechnik

Rail-Editor: Technische Aspekte I

- *QT* für Programmierung der grafischen Benutzeroberfläche
- Signal- und Slottechnik
- Eventverarbeitung für Maus- und Tastendrücke

Rail-Editor: Technische Aspekte I

- *QT* für Programmierung der grafischen Benutzeroberfläche
- Signal- und Slottechnik
- Eventverarbeitung für Maus- und Tastendrücke
- Basisklassen abgeleitet und Funktionalitäten erweitert

Rail-Editor: Technische Aspekte I

- *QT* für Programmierung der grafischen Benutzeroberfläche
- Signal- und Slottechnik
- Eventverarbeitung für Maus- und Tastendrücke
- Basisklassen abgeleitet und Funktionalitäten erweitert
- Graphstruktur für Syntax-Highlighting
  - internes Backend

Rail-Editor: Technische Aspekte I

- *QT* für Programmierung der grafischen Benutzeroberfläche
- Signal- und Slottechnik
- Eventverarbeitung für Maus- und Tastendrücke
- Basisklassen abgeleitet und Funktionalitäten erweitert
- Graphstruktur für Syntax-Highlighting
  - internes Backend
- Smart-Cursor und Grab-Modus
  - für intuitives Schreiben von Quellcode
  - siehe Live-Demo

Rail-Editor: Technische Aspekte II

- Main-Window als *Brain*
  - Weiterleitung an Child-Widgets

Rail-Editor: Technische Aspekte II

- Main-Window als *Brain*
  - Weiterleitung an Child-Widgets
- Undo-Redo-Funktionalität
  - abstrakte Klasse
  - wird durch konkrete Aktionen implementiert

Rail-Editor: Technische Aspekte II

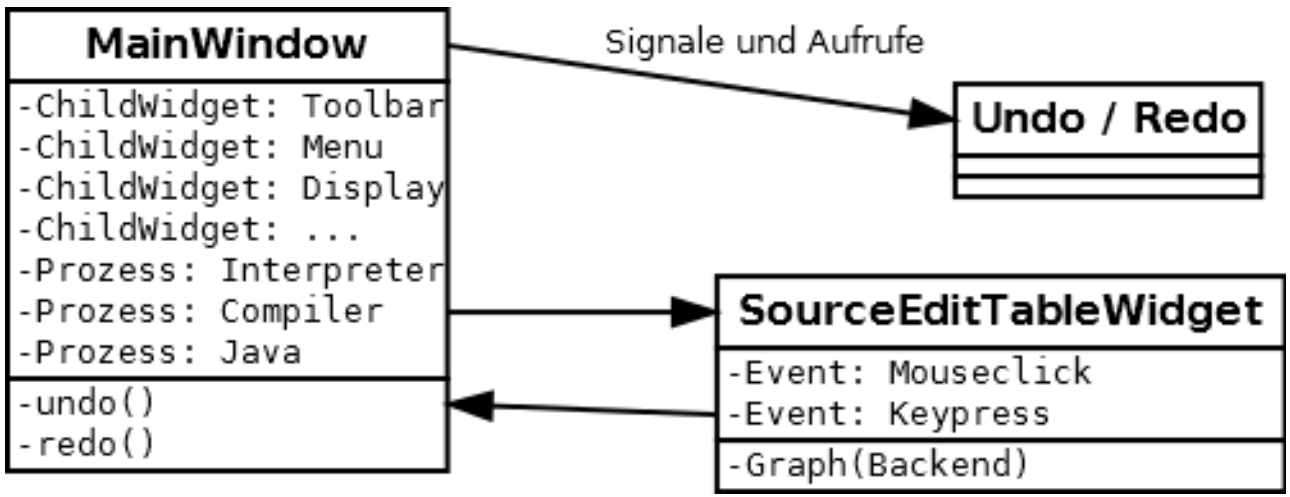
- Main-Window als *Brain*
  - Weiterleitung an Child-Widgets
- Undo-Redo-Funktionalität
  - abstrakte Klasse
  - wird durch konkrete Aktionen implementiert
- Compiler-Einbindung
  - Funktionen: Build, Run, Stop
  - auch der Rail-Interpreter kann verwendet werden

Rail-Editor: Technische Aspekte II

- Main-Window als *Brain*
  - Weiterleitung an Child-Widgets
- Undo-Redo-Funktionalität
  - abstrakte Klasse
  - wird durch konkrete Aktionen implementiert
- Compiler-Einbindung
  - Funktionen: Build, Run, Stop
  - auch der Rail-Interpreter kann verwendet werden
- *Preferences* (Editor-Einstellungen)
  - persistent gespeichert

Rail-Editor: Technische Aspekte II

- Main-Window als *Brain*
  - Weiterleitung an Child-Widgets
- Undo-Redo-Funktionalität
  - abstrakte Klasse
  - wird durch konkrete Aktionen implementiert
- Compiler-Einbindung
  - Funktionen: Build, Run, Stop
  - auch der Rail-Interpreter kann verwendet werden
- *Preferences* (Editor-Einstellungen)
  - persistent gespeichert



Outline

Make Titles Informative. Use Uppercase Letters. Long Titles are Split Automatically.

Freie Universität Berlin

- Use `itemize` a lot.
- Use very short sentences or short phrases.

Make Titles Informative.

You can create overlays. . .

- using the pause command:
  - First item.

Make Titles Informative.

You can create overlays. . .

- using the pause command:
  - First item.
  - Second item.
- using overlay specifications:

- using the general uncover command:

Make Titles Informative.

You can create overlays. . .

- using the pause command:
  - First item.
  - Second item.
- using overlay specifications:
  - First item.

- using the general uncover command:

Make Titles Informative.

You can create overlays. . .

- using the pause command:
  - First item.
  - Second item.
- using overlay specifications:
  - First item.
  - Second item.
- using the general uncover command:

Make Titles Informative.

You can create overlays. . .

- using the pause command:
  - First item.
  - Second item.
- using overlay specifications:
  - First item.
  - Second item.
- using the general uncover command:
  - First item.

Make Titles Informative.

You can create overlays. . .

- using the pause command:
  - First item.
  - Second item.
- using overlay specifications:
  - First item.
  - Second item.
- using the general uncover command:
  - First item.
  - Second item.

Outline

An old algorithm

```
int main (void)
{
  std::vector<bool> is_prime (100, true);
  for (int i = 2; i < 100; i++)
    if (is_prime[i])
      {
        std::cout << i << " ";
        for (int j = i; j < 100;
             is_prime [j] = false, j+=i);
      }
  return 0;
}
```

An Algorithm For Finding Primes Numbers.

```
int main (void)
{
 std::vector<bool> is_prime (100, true);
 for (int i = 2; i < 100; i++)




 return 0;
}
```

An Algorithm For Finding Primes Numbers.

```
int main (void)
{
 std::vector<bool> is_prime (100, true);
 for (int i = 2; i < 100; i++)
 if (is_prime[i])
 {



 }
 return 0;
}
```

An Algorithm For Finding Primes Numbers.

```
int main (void)
{
 std::vector<bool> is_prime (100, true);
 for (int i = 2; i < 100; i++)
 if (is_prime[i])
 {
 std::cout « i « " ";
 for (int j = i; j < 100;
 is_prime [j] = false, j+=i);
 }
 return 0;
}
```

An Algorithm For Finding Primes Numbers.

```
int main (void)
{
 std::vector<bool> is_prime (100, true);
 for (int i = 2; i < 100; i++)
 if (is_prime[i])
 {
 std::cout « i « " ";
 for (int j = i; j < 100;
 is_prime [j] = false, j+=i);
 }
 return 0;
}
```

Note the use of `std::`.

Outline

Make Titles Informative.

Example

- 2 is prime (two divisors: 1 and 2).
- 3 is prime (two divisors: 1 and 3).
- 4 is not prime (three divisors: 1, 2, and 4).

Make Titles Informative.

Theorem

There is no largest prime number and, in addition,

$$\int_\Omega \nabla u \cdot \nabla v = -\int_\Omega u \Delta v + \int_{\partial\Omega} uvn$$

Proof.

1. Suppose p were the largest prime number.

4. Thus $q+1$ is also prime and greater than p. □

Make Titles Informative.

Theorem

There is no largest prime number and, in addition,

$$\int_\Omega \nabla u \cdot \nabla v = -\int_\Omega u \Delta v + \int_{\partial\Omega} uvn$$

Proof.

1. Suppose p were the largest prime number.
2. Let q be the product of the first p numbers.

4. Thus $q+1$ is also prime and greater than p. □

Make Titles Informative.

Theorem

There is no largest prime number and, in addition,

$$\int_\Omega \nabla u \cdot \nabla v = -\int_\Omega u \Delta v + \int_{\partial\Omega} uvn$$

Proof.

1. Suppose p were the largest prime number.
2. Let q be the product of the first p numbers.
3. Then $q+1$ is not divisible by any of them.
4. Thus $q+1$ is also prime and greater than p. □

Make Titles Informative.

Theorem

There is no largest prime number and, in addition,

$$\int_\Omega \nabla u \cdot \nabla v = -\int_\Omega u \Delta v + \int_{\partial\Omega} uvn$$

Proof.

1. Suppose p were the largest prime number.
2. Let q be the product of the first p numbers.
3. Then $q+1$ is not divisible by any of them.
4. Thus $q+1$ is also prime and greater than p. □

The proof used *reductio ad absurdum*.

Make Titles Informative.

Outline

Make Titles Informative.

Make Titles Informative.

Make Titles Informative.

Summary

- The first main message of your talk in one or two lines.
- The second main message of your talk in one or two lines.
- Perhaps a third message, but not more than that.

- Outlook
  - Something you haven’t solved.
  - Something else you haven’t solved.

For Further Reading I

A. Author.
Handbook of Everything.
Some Press, 1990.

S. Someone.
On this and that.
Journal of This and That, 2(1):50–100, 2000.