Wissenschaftliche Softwareentwicklung

4. Editoren/IDEs

Till Biskup
Physikalische Chemie
Universität Rostock
03.11.2023

Zentrale Aspekte

- Programmierung bedeutet (meist) das Erzeugen von reinem Text.
  Im Prinzip ist also jeder Texteditor geeignet.
- Programmierer verbringen viel Zeit mit dem Editor.
  Die Wahl kann für die Produktivität entscheidend sein.
- Moderne Editoren bringen Funktionalität mit,
  die die Programmierung erleichtert (und Fehlern vorbeugt).
- IDEs integrieren viele Werkzeuge in einer Oberfläche.
  Die Komplexität bedingt eine steile initiale Lernkurve.
- IDEs ersetzen nicht die solide Kenntnis des Umgangs
  mit den grundlegenden Programmierwerkzeugen.

Übersicht

Motivation: Warum ist die Wahl des Editors wichtig?

Mindestanforderungen an einen Editor/eine IDE

Editoren vs. Entwicklungsumgebungen (IDEs)

Warum ist die Wahl des Editors wichtig?

Der Programmierer verbringt die meiste Zeit mit seinem Editor.

Gründe für eine sorgfältige Wahl des Editors

- Programme bestehen (meist) aus reinem Text.
  - Im Prinzip ist jeder Texteditor geeignet.
  - Editoren zum Programmieren sind Texteditoren.
- Programmierer verbringen die meiste Zeit mit dem Editor.
  - Das Programm kann die Produktivität entscheidend beeinflussen.
  - Vertrautheit mit dem verwendeten Editor ist entscheidend.
- Code wird viel häufiger gelesen als geschrieben.
  - Gute Editoren erleichtern (erheblich) die Lesbarkeit und erhöhen dadurch die Produktivität.
- Gute Editoren steigern die Codequalität
  - Unterstützung von Refactoring und automatisierten Tests

☛ Tipp: sorgfältig ausprobieren und bewusst entscheiden

Warum ist die Wahl des Editors wichtig?

Der Editor ist das wichtigste Werkzeug des Programmierers.

Der Wert guter Werkzeuge

> *Every craftsman starts his or her journey with a basic set of good-quality tools. [...]*
>
> *Tools amplify your talent.*
> *The better your tools,*
> *and the better you know how to use them,*
> *the more productive you can be.*
> *Start with a basic set of generally applicable tools.*
>
> – Andrew Hunt, David Thomas

- Der Editor ist das wichtigste Werkzeug des Programmierers.

Andrew Hunt, David Thomas: The Pragmatic Programmer, Addison-Wesley, Boston 1999, S. 71

Übersicht

Grundlegende Eigenschaften guter Editoren

- (komplett) mit der Tastatur bedienbar
  - Hände können immer auf der Tastatur bleiben.
  - beschleunigt (erheblich) das Arbeiten
- konfigurierbar
  - Jeder hat andere Vorstellungen und Vorlieben.
  - Bsp.: Schriftarten, Farben, Fenstergrößen, Tastaturkürzel
- erweiterbar
  - Unterstützung weiterer (Programmier-)Sprachen
  - Integration mit beliebigen Compilern
- programmierbar
  - Automatisierung wiederkehrender Abläufe
  - Möglichkeiten: Makros oder (eingebaute) Skriptsprachen

Spezifische Eigenschaften für jede Programmiersprache

- Syntaxhervorhebung
  - erhöht (wesentlich) die Lesbarkeit
- Autovervollständigung
  - beugt (konsequent genutzt) Tippfehlern vor
  - erspart mitunter den Blick in die Dokumentation
- automatische Einrückung
  - konsistentes Erscheinungsbild
  - wichtig: sollte konfigurierbar sein
- Vorlagen/Textbausteine
  - spart Tipparbeit (Beschleunigung) und sorgt für Konsistenz
- Refactoring
  - erhöht wesentlich die Lesbarkeit und Qualität von Code

Listing: Beispiel mit Syntax-Hervorhebung

```
    def redo(self):
        """Reapply previously undone processing step.

        Raises
        ------
        RedoAlreadyAtLatestChangeError
            Raised when trying to redo with empty history
        """
        if self._history_pointer == len(self.history) - 1:
            raise RedoAlreadyAtLatestChangeError
        processing_step_record = \
            self.history[self._history_pointer + 1].processing
        processing_step = processing_step_record.create_processing_step()
        processing_step.process(self)
        self._increment_history_pointer()

    def _has_leading_history(self):
        if len(self.history) - 1 > self._history_pointer:
            return True
        else:
            return False
```

Übersicht

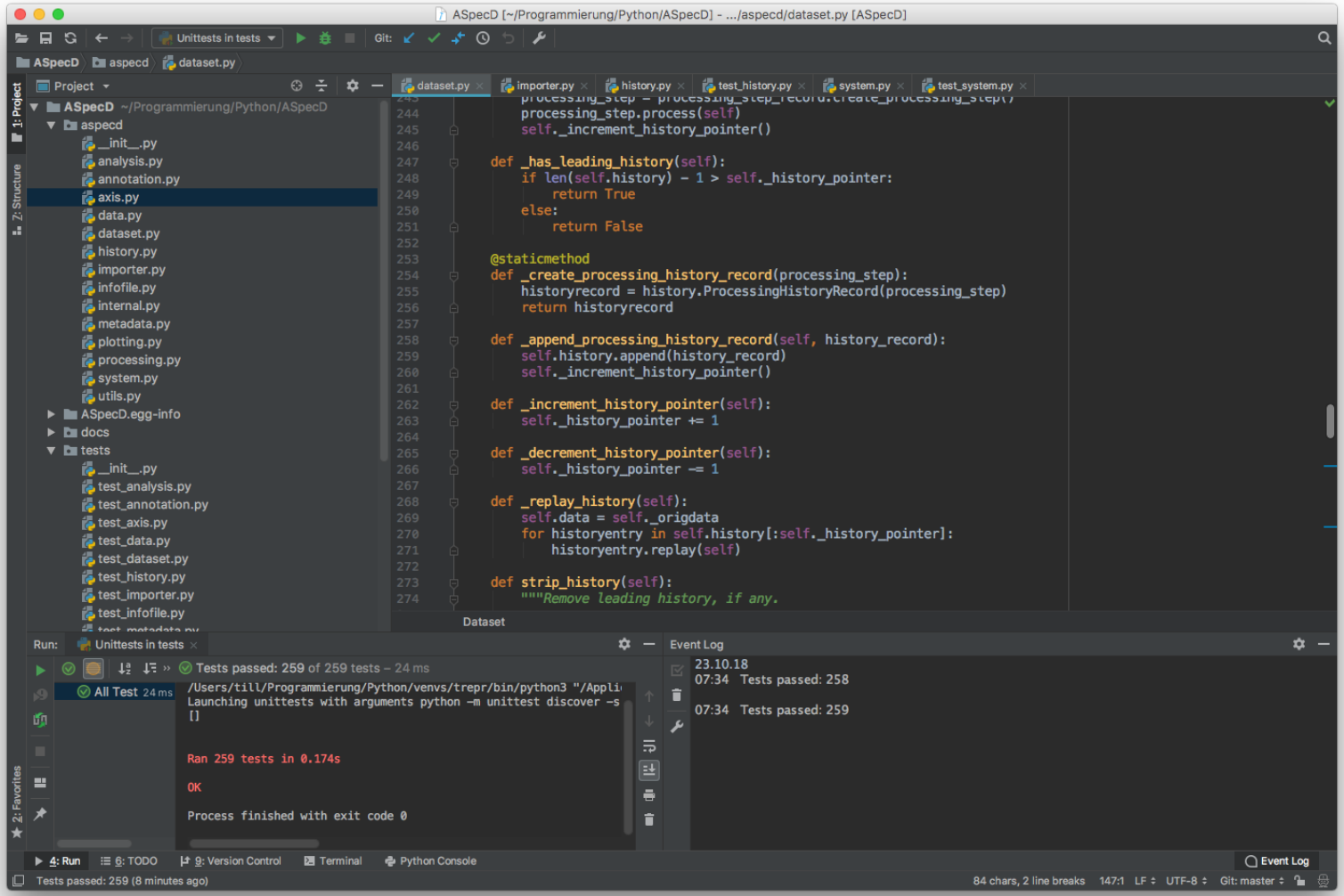
integrierte Entwicklungsumgebung

engl. *integrated development environment* (IDE),
gemeinsame Oberfläche für Werkzeuge zur Programmierung,
um Softwareentwicklung ohne Medienbrüche zu ermöglichen

- eine Oberfläche für (fast) alle Aufgaben
  - Editor, Compiler, Interpreter, Debugger, VCS, ...
  - Hilfe/Dokumentation, Steuerung externer Services, ...
- Integration existierender Werkzeuge
  - Compiler, Interpreter, Debugger, ...
  - meist weitestgehend konfigurierbar

Editoren vs. IDEs

Was unterscheidet eine IDE von einem Editor?



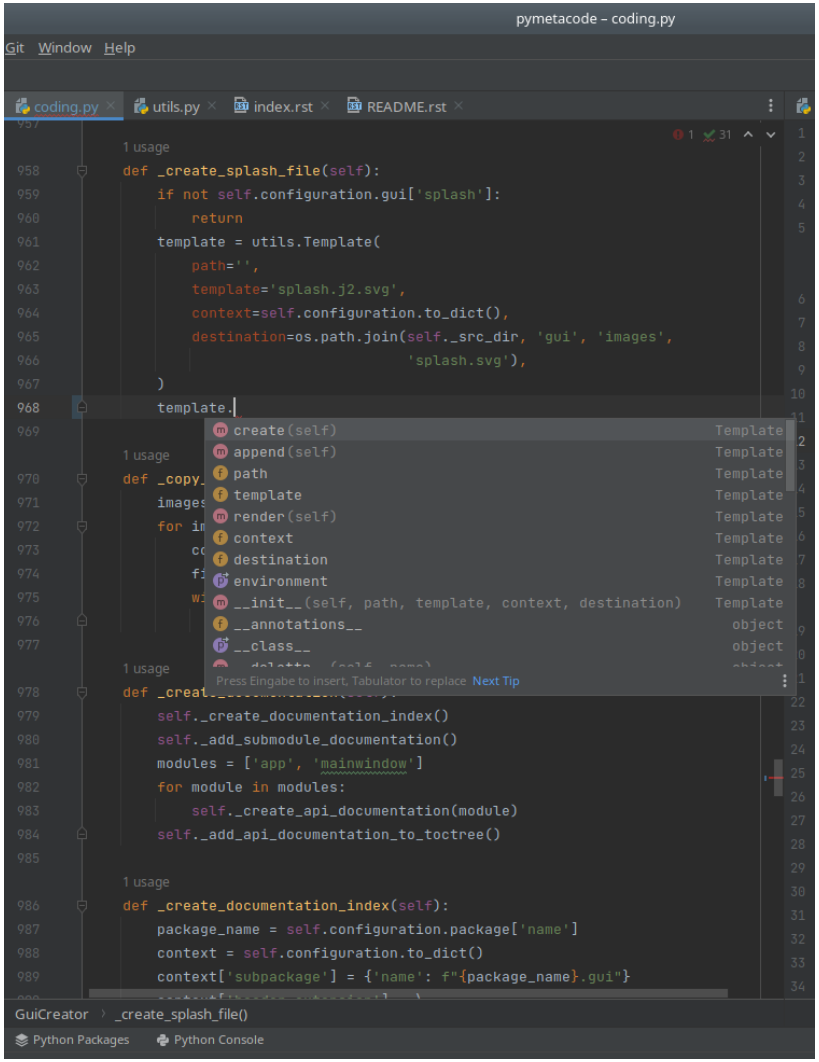
Einige zusätzliche Fähigkeiten von IDEs

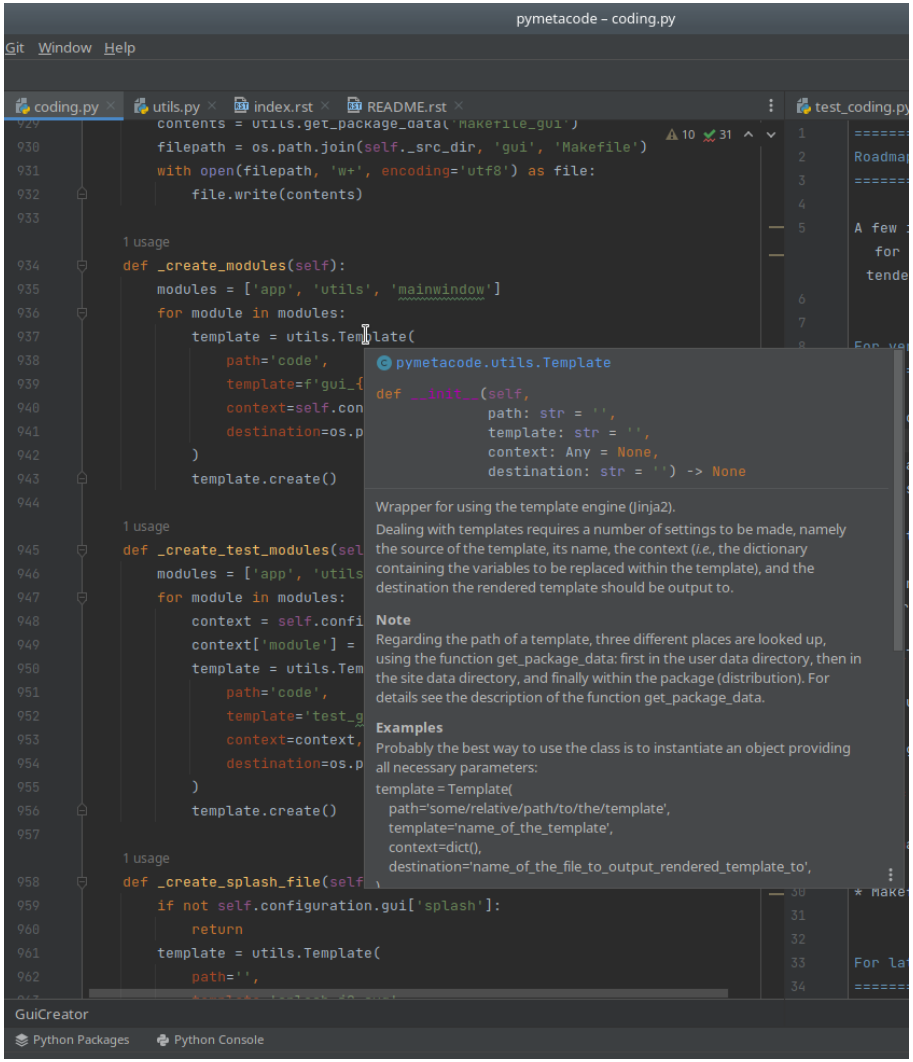
- Refactoring
  - komplexe Ersetzungen/Umbenennungen, Modularisierung, ...
- Integration der Buildumgebung
  - Compiler, Linker, ggf. Interpreter
- Debugger
  - interaktive Fehlerdetektion und Behebung
- Hilfe
  - kontextspezifisch und Sprachdokumentation
- Integration weiterer externer Komponenten
  - Bsp.: VCS, Datenbanken, Webserver; Tests

☛ Grenze zwischen Editor und IDE mitunter fließend

Editoren vs. IDEs

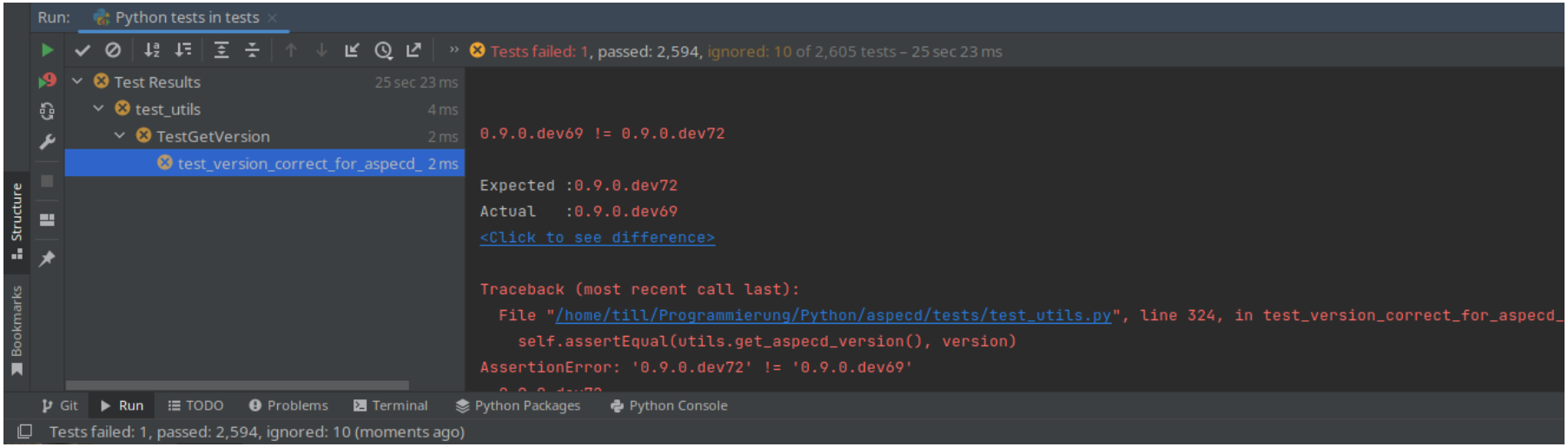
Beispiel: Autovervollständigung von Code und Dokumentationsvorschau

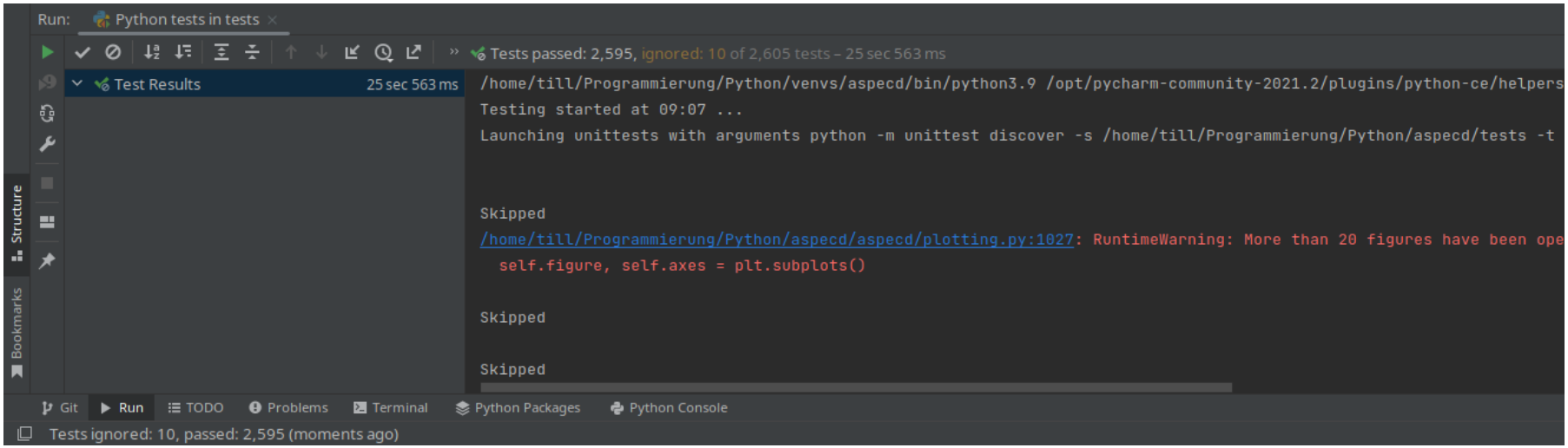




Editoren vs. IDEs

Beispiel: Integration von Unittests





Editoren vs. IDEs

Vor- und Nachteile von IDEs gegenüber Editoren

Vorteile von IDEs

- alles aus einer gemeinsamen Oberfläche erreichbar
- Beschleunigung und Vereinfachung von Abläufen
- manche Aspekte schwer in reinem Editor realisierbar

Nachteile von IDEs

- steile (initiale) Lernkurve
- verstecken sehr viel vor dem Nutzer

☛ Kosten-Nutzen-Abwägung, abhängig von
  - Umfang und Dauer des Projekts
  - Erfahrung des Nutzers

Editoren vs. IDEs

IDEs sind nicht die ganze Welt – auch wenn es so scheint.

> *Many new programmers make the mistake*
> *of adopting a single power tool,*
> *such as a particular integrated development environment (IDE),*
> *and never leave its cozy interface.*
>
> *This really is a mistake.*
> *We need to be comfortable beyond the limits*
> *imposed by an IDE. The only way to do this is to*
> *keep the basic tool set sharp and ready to use.*
>
> – Andrew Hunt, David Thomas

- Nichts kann die solide Kenntnis des Umgangs mit grundlegenden Programmierwerkzeugen ersetzen.

Andrew Hunt, David Thomas: The Pragmatic Programmer, Addison-Wesley, Boston 1999, S. 72

Zentrale Aspekte

- Programmierung bedeutet (meist) das Erzeugen von reinem Text.
  Im Prinzip ist also jeder Texteditor geeignet.
- Programmierer verbringen viel Zeit mit dem Editor.
  Die Wahl kann für die Produktivität entscheidend sein.
- Moderne Editoren bringen Funktionalität mit,
  die die Programmierung erleichtert (und Fehlern vorbeugt).
- IDEs integrieren viele Werkzeuge in einer Oberfläche.
  Die Komplexität bedingt eine steile initiale Lernkurve.
- IDEs ersetzen nicht die solide Kenntnis des Umgangs
  mit den grundlegenden Programmierwerkzeugen.