Freie Universität Berlin

Stile von API-Dokumentationen anhand von Java und Python

Betreuer & Erstgutachter: Prof. Dr. Lutz Prechelt
Zweitgutachterin: Prof. Dr. Claudia Müller-Birn

Masterarbeit Verteidigung

Ahmet-Serdar Karakaya
ahmetserdar@hotmail.de

22.05.2017

Freie Universität Berlin
Institut für Informatik
Arbeitsgruppe Software Engineering

Gliederung

- Einleitung
- Wissenstypen in API-Dokumentationen
- Forschungsmethoden
- Stileigenschaften und Dokumentationsstile
- Zusammenfassung und Ausblick

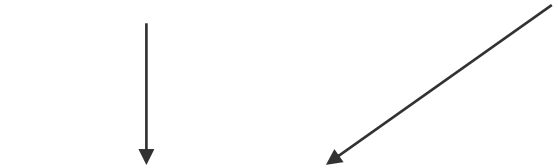
Einleitung

- **Was ist eine API?**
- API = Application Programming Interface
- Programmierschnittstelle zu Softwarebibliothek
- Beispiele:
  - Java™ Platform, Standard Edition 6 (7, 8, ...)
  - The Python Standard Library
  - Google APIs (Drive API, Translation API, …)
  - Dropbox API
  - …

Einleitung

- **Warum eine API nutzen?**

```java
int[] intArray = { 4, 1, 5, 2, 7 };
Arrays.sort(intArray);
System.out.println(Arrays.toString(intArray));
```



[1, 2, 4, 5, 7]

```java
public static void main(String[] args) {

    int[] intArray = { 4, 1, 5, 2, 7 };
    int low = 0;
    int high = intArray.length - 1;
    quickSort(intArray, low, high);
    System.out.println(Arrays.toString(intArray));
}
//quicksort
public static void quickSort(int[] intArray, int low, int high) {

    int middle = low + (high - low) / 2;
    int pivot = intArray[middle];
    int i = low, j = high;
    while (i <= j) {
        while (intArray[i] < pivot) {
            i++;
        }

        while (intArray[j] > pivot) {
            j--;
        }

        if (i <= j) {
            int temp = intArray[i];
            intArray[i] = intArray[j];
            intArray[j] = temp;
            i++;
            j--;
        }
    }

    if (low < j)
        quickSort(intArray, low, j);

    if (high > i)
        quickSort(intArray, i, high);
}
```

Einleitung

- APIs bieten viele Funktionalitäten an ⟹ APIs sind groß und komplex (Ⅰ)
- APIs helfen bei der Softwareentwicklung ⟹ APIs sind weit verbreitet (Ⅱ)

  Ⅰ + Ⅱ ⟹ *Benutzerfreundlichkeit* einer API (API-Usability) ist wichtig

- Benutzerfreundlichkeit: Wie intuitiv ist die Bedienung eines Systems?
- Bei APIs:
  - Wie gut/schnell kann ich die API lernen?
  - Wie gut/schnell kann ich ein Problem lösen?

Einleitung

- APIs bieten viele Funktionalitäten an ⟹ APIs sind groß und komplex (Ⅰ)
- APIs helfen bei der Softwareentwicklung ⟹ APIs sind weit verbreitet (Ⅱ)

  Ⅰ + Ⅱ ⟹ *Benutzerfreundlichkeit* einer API (API-Usability) ist wichtig

- Benutzerfreundlichkeit: Wie intuitiv ist die Bedienung eines Systems?
- Bei APIs:
  - Wie gut/schnell kann ich die API lernen?
  - Wie gut/schnell kann ich ein Problem lösen?

  } API-Dokumentation!

Einleitung

- „Die meisten [1, 2] bzw. dramatischsten [3] bzw. fundamentalsten [4] API-Usability-Probleme rühren von der API-Dokumentation her.“ [5]

- Es müssen bessere API-Dokumentationen her

  - „Entwurfsmuster für API-Dokumentationen“ können Autoren helfen

Gliederung

- Einleitung ✓

➢ Wissenstypen in API-Dokumentationen

Wissenstypen in API-Dokumentationen

- Maalej und Robillard ermitteln 12 „Wissenstypen“ (*Knowledge Type*) [6]
- Wissenstyp: Welcher Typ von Wissen oder Information ein Textabschnitt vermittelt
- Functionality & Behavior: *Was* macht ein API-Element?
- Directives: Was mit den API-Elementen (nicht) gemacht werden darf
- Structure & Relationship: Struktur von Klassen & Beziehungen innerhalb der API
- Environment: Betriebssystem, Hardware, Version, …

Wissenstypen in API-Dokumentationen

`socket.setsockopt`(*level*, *optname*, *value*)

Set the value of the given socket option (see the Unix manual page *setsockopt(2)*). The needed symbolic constants are defined in the `socket` module (`SO_*` etc.). The value can be an integer or a bytes-like object representing a buffer. In the latter case it is up to the caller to ensure that the bytestring contains the proper bits (see the optional built-in module `struct` for a way to encode C structures as bytestrings).

Changed in version 3.5: Writable bytes-like object is now accepted.

Legende:

- Functionality & Behavior
- References
- Structure & Relationship
- Directives
- Environment

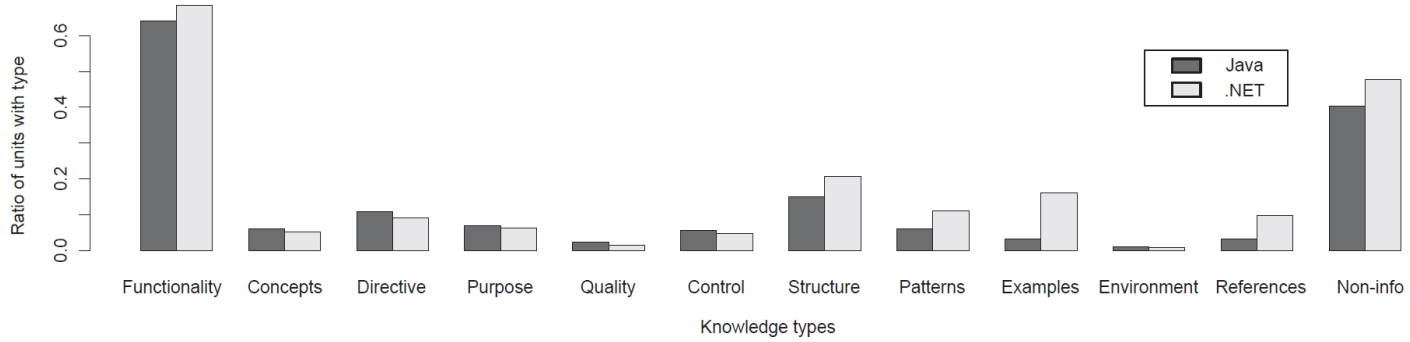
Wissenstypen in API-Dokumentationen

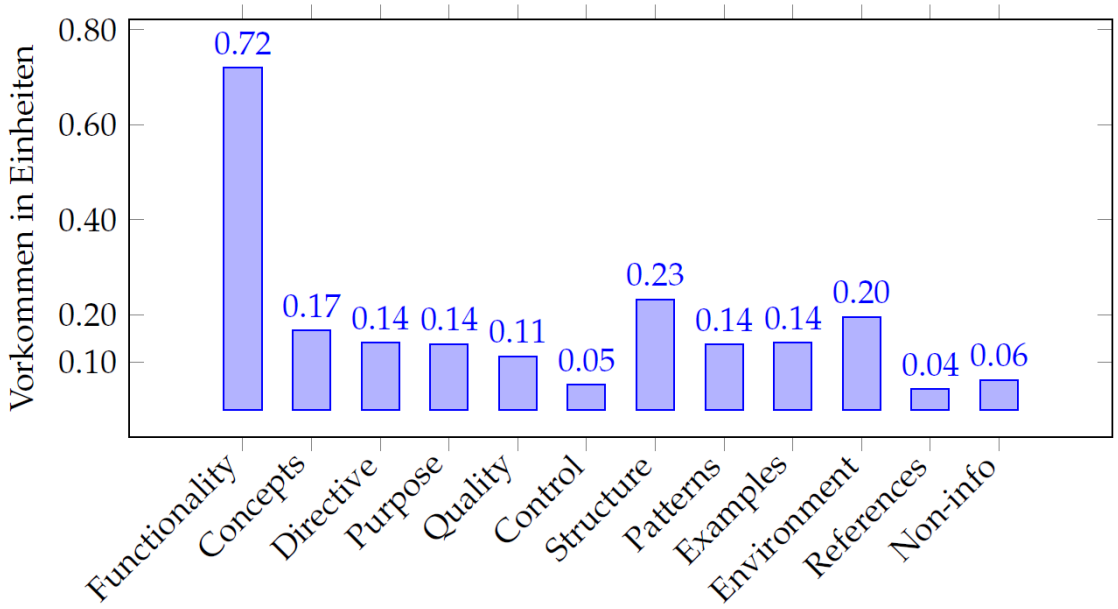
- Maalej und Robillard: Analyse der Java- und .NET 4.0-API-Dokumentationen
- Zufällige Textabschnitte aus der Dokumentation
- Vorkommende Wissenstypen markiert

Wissenstypen in API-Dokumentationen

- Wildermann: Analyse der Python-API-Dokumentation [7]
- Zufällige Textabschnitte aus der Dokumentation
- Abschnitte gemäß ihrem Wissenstyp markiert
- Dadurch sind genauere Aussagen möglich

Wissenstypen in API-Dokumentationen





Gliederung

- Einleitung ✓
- Wissenstypen in API-Dokumentationen ✓
- ➢ Forschungsmethoden

Forschungsmethoden

- Dokumentation der Java- und der Python-API unterschiedlich
- Beide API-Dokumentationen sind gut
- Hypothese: Es gibt unterschiedliche Stile von API-Dokumentationen, die zu guten API-Dokumentationen mit unterschiedlichen Vor- und Nachteilen führen.

Forschungsmethoden

- Hypothese: Es gibt unterschiedliche Stile von API-Dokumentationen, die zu guten API-Dokumentationen mit unterschiedlichen Vor- und Nachteilen führen.
- Wie untersuchen???
- Mögliche Ansätze:
  - Methode von Wildermann nochmal für Java
  - Grounded Theory Methodologie
  - Typisierung von ganzen Dokumentationsseiten anstatt zufälligen Blöcken

Forschungsmethoden

- Hypothese: Es gibt unterschiedliche Stile von API-Dokumentationen, die zu guten API-Dokumentationen mit unterschiedlichen Vor- und Nachteilen führen.

- Wie untersuchen???

- Mögliche Ansätze:
  - Methode von Wildermann nochmal für Java

  - Grounded Theory Methodologie

  ➢ Typisierung von ganzen Dokumentationsseiten anstatt zufälligen Blöcken

Forschungsmethoden

- Typisierung von ganzen Dokumentationsseiten anstatt zufälligen Blöcken
- Welche Dokumentationsseiten vergleichen?
- Dokumentationsseiten zu Klassen mit ähnlichen Funktionen
- Denn: Ähnliche Funktionalitäten müssen ähnlich beschrieben sein
- Sonst: Unterschiedliche Stile!
- Problem: Solche passenden Pendants gibt es nicht…

Forschungsmethoden

- Problem: Solche passenden Pendants gibt es nicht…
- Idee: Pendants selber erstellen
- Pendants typisiert und verglichen
- Problem: keine offensichtlichen relevanten Unterschiede

Forschungsmethoden

- Andere Methoden versucht:
  - Grounded Theory
  - E-Mail-Verteiler von Autoren
  - Diskussionsrunde mit „Python Users Berlin“
  - Immer wieder mit Wissenstypen beschäftigt

Gliederung

- Einleitung ✓
- Wissenstypen in API-Dokumentationen ✓
- Forschungsmethoden ✓

➢ Dokumentationsstile und Stileigenschaften

Dokumentationsstile und Stileigenschaften

- Zwei API-Dokumentationsstile:
  - Referenzstil
  - Buchstil

- Fünf Stileigenschaften:
  - Aufbau und Form der Dokumentation
  - Beispielnutzung
  - Lokalität
  - Beziehungssichtbarkeit
  - Wissenstypenordnung

Dokumentationsstile und Stileigenschaften

- Aufbau und Form der Dokumentation
  - Bei Java:

    http://docs.oracle.com/javase/6/docs/api/index.html
    - Struktur wie die der API
    - Homogener Aufbau der Dokumentationsseiten
    - Wird aus dem Quelltext der API generiert

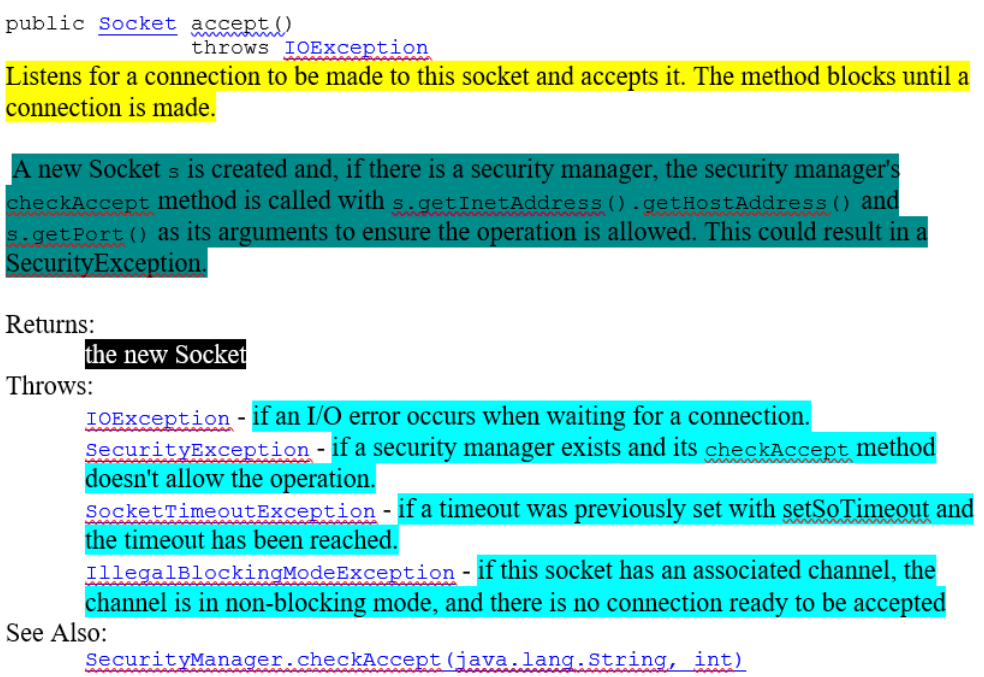
  - Bei Python:

    https://docs.python.org/3.4/library/
    - Didaktische Struktur
    - Keine fest vorgegebene Form der Dokumentationsseiten
    - Wird extra erstellt und gepflegt

Dokumentationsstile und Stileigenschaften

- Wissenstypenordnung
  - Bei Java:
  - Pro Satz/Textblock nur ein Wissenstyp
  - Ist eine Tendenz

  + Uninteressante Abschnitte können besser übersprungen werden
  − Langweiliger beim vollständigen Durchlesen



```
public Socket accept()
                throws IOException
```

Listens for a connection to be made to this socket and accepts it. The method blocks until a connection is made.

A new Socket `s` is created and, if there is a security manager, the security manager's `checkAccept` method is called with `s.getInetAddress().getHostAddress()` and `s.getPort()` as its arguments to ensure the operation is allowed. This could result in a SecurityException.

Returns:

the new Socket

Throws:

`IOException` - if an I/O error occurs when waiting for a connection.

`SecurityException` - if a security manager exists and its `checkAccept` method doesn't allow the operation.

`SocketTimeoutException` - if a timeout was previously set with setSoTimeout and the timeout has been reached.

`IllegalBlockingModeException` - if this socket has an associated channel, the channel is in non-blocking mode, and there is no connection ready to be accepted

See Also:

`SecurityManager.checkAccept(java.lang.String, int)`

Dokumentationsstile und Stileigenschaften

- Wissenstypenordnung
  - Bei Python:
  - Starke Durchmischung von Wissenstypen

  + Mehr Freiheit für den Autor
  + Besserer Lesefluss
  − Schwer zu überfliegen

classmethod `datetime.fromtimestamp`(*timestamp*, *tz*=*None*)

Return the local date and time corresponding to the POSIX timestamp, such as is returned by `time.time()`. If optional argument *tz* is `None` or not specified, the timestamp is converted to the platform's local date and time, and the returned `datetime` object is naive.

If *tz* is not `None`, it must be an instance of a `tzinfo` subclass, and the timestamp is converted to *tz*'s time zone. In this case the result is equivalent to `tz.fromutc(datetime.utcfromtimestamp(timestamp).replace(tzinfo=tz))`.

`fromtimestamp()` may raise `OverflowError`, if the timestamp is out of the range of values supported by the platform C `localtime()` or `gmtime()` functions, and `OSError` on `localtime()` or `gmtime()` failure. It's common for this to be restricted to years in 1970 through 2038. Note that on non-POSIX systems that include leap seconds in their notion of a timestamp, leap seconds are ignored by `fromtimestamp()`, and then it's possible to have two timestamps differing by a second that yield identical `datetime` objects. See also `utcfromtimestamp()`.

Changed in version 3.3: Raise `OverflowError` instead of `ValueError` if the timestamp is out of the range of values supported by the platform C localtime() or gmtime() functions. Raise `OSError` instead of `ValueError` on localtime() or gmtime() failure.

Dokumentationsstile und Stileigenschaften

- Referenzstil (Java):
  - Gut für gezielte Informationssuche (Wenn Bezeichnung bekannt)
  - Schlecht zum Lernen der API
  - Weniger Aufwand in der Erstellung und Wartung

- Buchstil (Python):
  - Gut zum Lernen der API
  - Schlecht für schnelle und gezielte Informationssuche
  - Mehr Aufwand in der Erstellung und Wartung

Gliederung

- Einleitung ✓
- Wissenstypen in API-Dokumentationen ✓
- Forschungsmethoden ✓
- Dokumentationsstile und Stileigenschaften ✓

➢ Zusammenfassung und Ausblick

Zusammenfassung und Ausblick

- APIs sind weit verbreitet und wichtig
- API-Usability ist wichtig
- API-Usability wird stark von API-Dokumentation beeinflusst
- API-Dokumentationen sind allgemein schlecht
- „Entwurfsmuster für API-Dokumentationen“ können helfen
- Zwei Unterschiedliche API-Dokumentationsstile
- Fünf Stileigenschaften

Zusammenfassung und Ausblick

- Je nach Situation unterschiedliche Ausprägung

- Ausblick:
  - Weitere Dokumentationsstile
  - Weitere Stileigenschaften

  - Methoden:
    - Viele ganze Dokumentationsseiten typisieren
    - Mit Autoren sprechen
    - Distanz zu Wissenstypen und nochmal Grounded Theory
    - Andere API-Dokumentationen betrachten
      - Nach Wissenstypen typisieren
      - Stileigenschaften überprüfen

Vielen Dank für Ihre Aufmerksamkeit!

Noch Fragen?

Literaturverzeichnis

[1] T. Grill, O. Polacek, and M. Tscheligi, *Methods towards API Usability: A Structural Analysis of Usability Problem Categories*, pp. 164-180. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[2] M. F. Zibran, F. Z. Eishita, and C. K. Roy, *Useful, but usable? Factors affecting the usability of apis*, in Reverse Engineering (WCRE), 2011 18th Working Conference on, pp. 151-155, IEEE, 2011.

[3] M. P. Robillard and R. Deline, *A field study of api learning obstacles*, Empirical Softw. Eng., vol. 16, pp. 703-732, Dec. 2011.

[4] J. Bloch, *How to design a good api and why it matters*, in Companion to the 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications, OOPSLA '06, (New York, NY, USA), pp. 506-507, ACM, 2006.

[5] B. Kahlert, *API-Usability der auf Templatemetaprogrammierung basierenden Softwarebibliothek"SeqAn“*. PhD thesis, Freie Universtität Berlin,2015.

[6] W. Maalej and M. P. Robillard, *Patterns of knowledge in api reference documentation*, IEEE Transactions on Software Engineering, vol. 39, no. 9, pp. 1264-1282, 2013.

[7] S. Wildermann, *Messung der Informationstypen-Häugkeiten in der Python-Dokumentation*, Bachelor's thesis, Freie Universität Berlin, 2014.