Connascence

agenda

WHAT IS CONNASCENCE?

TYPES OF CONNASCENCE

BENEFITS OF CONNASCENCE

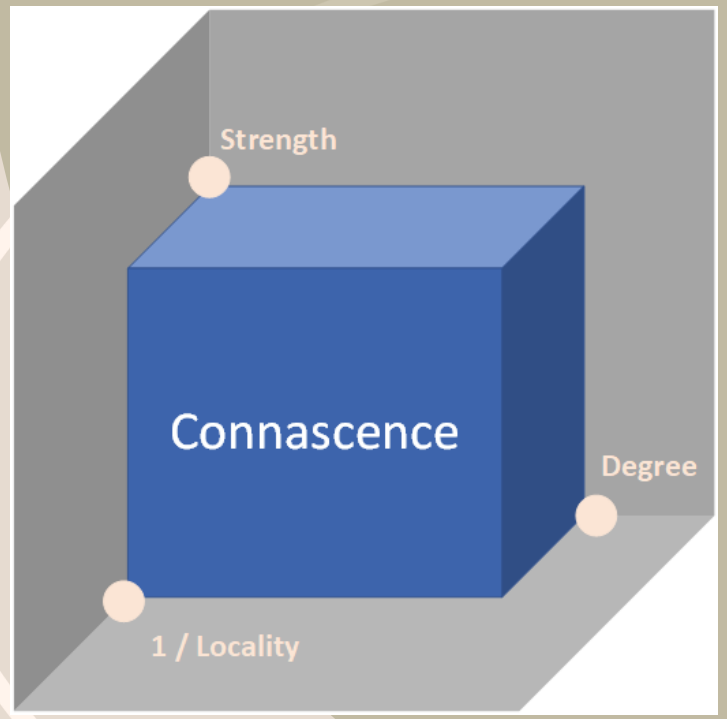
CLOSING THOUGHTS

What is connascence?

A metric to allow reasoning about the complexity caused by dependency in object-oriented design.

invented by Meilir Page-Jones

Two components are connascent if a change in one would require the other to be modified in order to maintain the overall correctness of the system.



What is connascence?

Strength

Stronger if more likely to require compensating changes in connascent elements

Degree

The degree of its occurrence - how many elements it affects

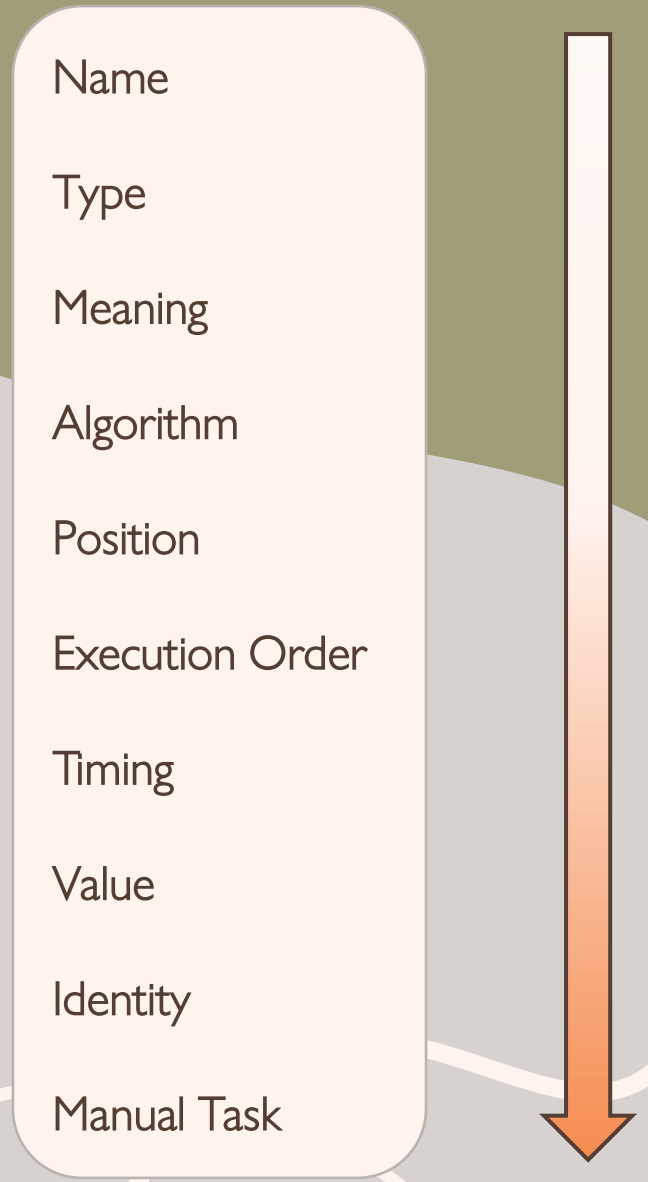
Might be acceptable to a small degree e.g. a function taking 2 args, but not so to a large degree, e.g a function taking 10 args

Locality

Stronger forms of connascence are acceptable if the elements involved are closely related.

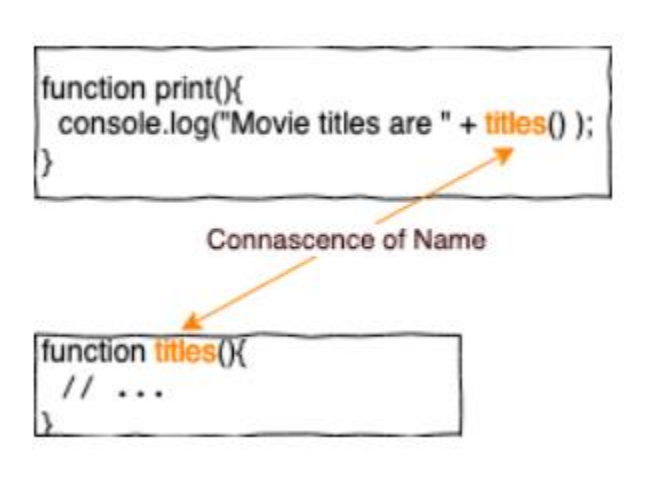
The same strength and degree of connascence will have a higher difficulty and cost of change, the more distant the involved elements are

Types of Connascence



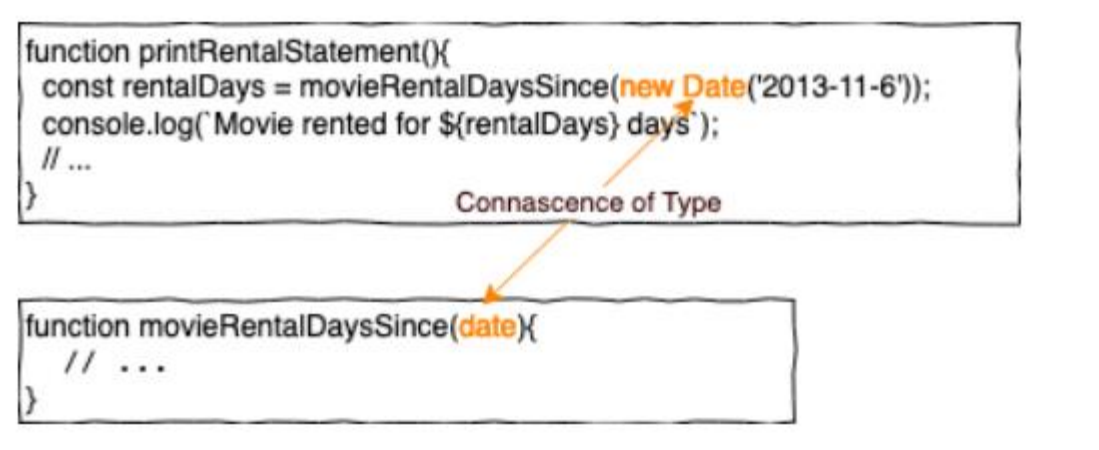
Connascence of Name

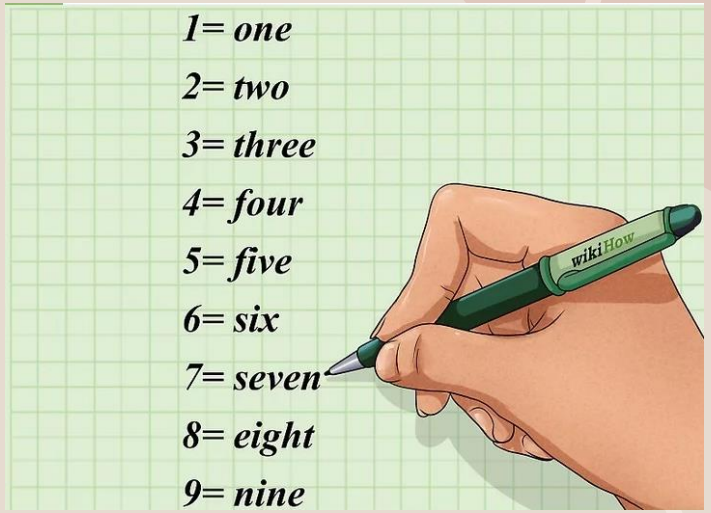
Multiple components must agree on the NAME of an entity, e.g. method names – when the method name changes anything calling that method will need to change



Connascence of Type

Multiple components must agree on the TYPE of an entity, e.g. If a method changes the type of its argument from an integer to a string – anything calling that method needs to pass in a different argument





Connascence of Meaning

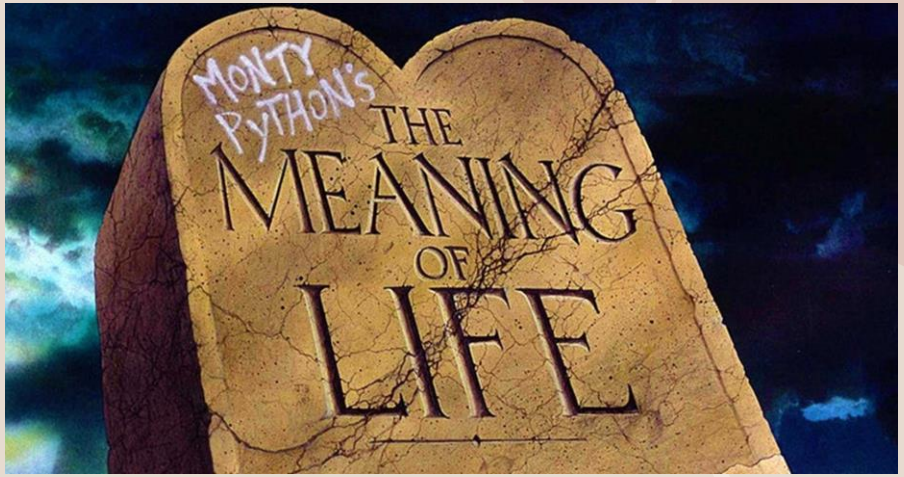
Multiple components must agree on the MEANING of particular values, e.g. returning integers 0 and 1 to represent false and true (also called connascensce of convention)

```
function printRentalStatement(){
  // ...
  let thisAmount = 0;
  switch(movie.code) {
    case "regular":
      thisAmount = 2; break;
    case "childrens":
      thisAmount = 1.5; break;
  }
  if(thisAmount > 25){
    // ...
  }
}
```

Connascence of Meaning

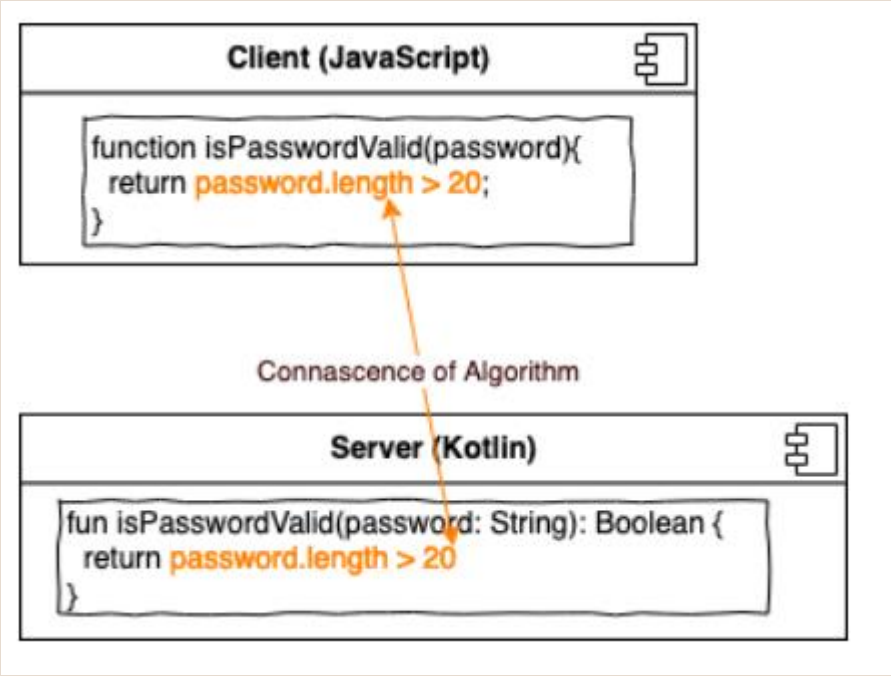
Connascence of Meaning

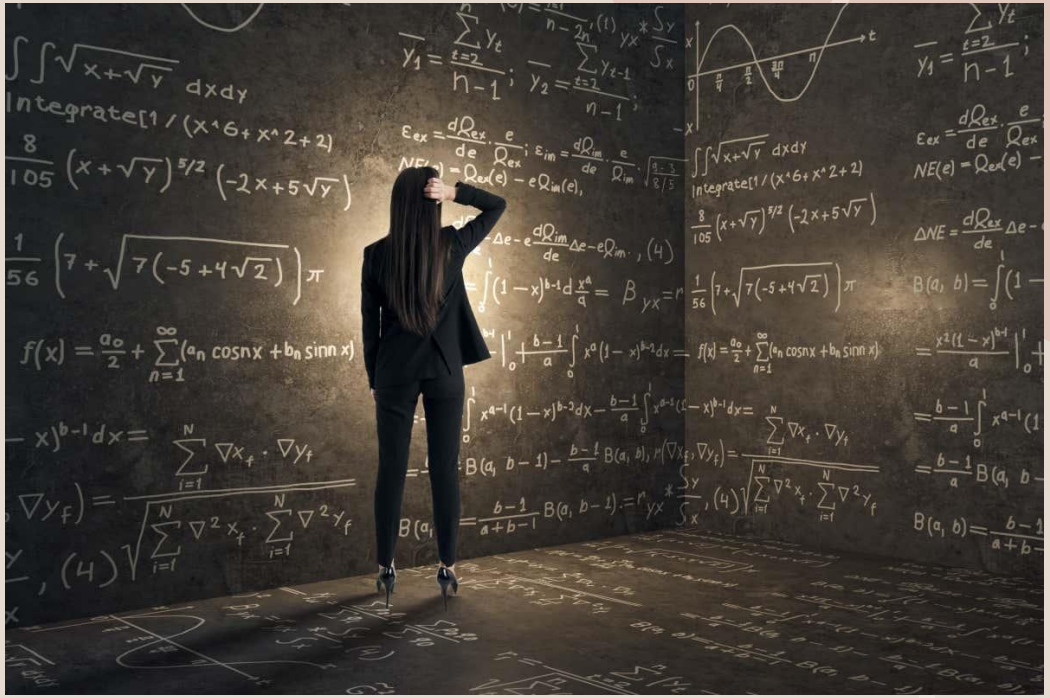
Connascence of Meaning



Connascence of Algorithm

Multiple components must agree on a PARTICULAR ALGORITHM, e.g. Message authentication codes - both sides of the exchange must implement exactly the same hashing algorithm or the authentication will fail.





Connascence of Position

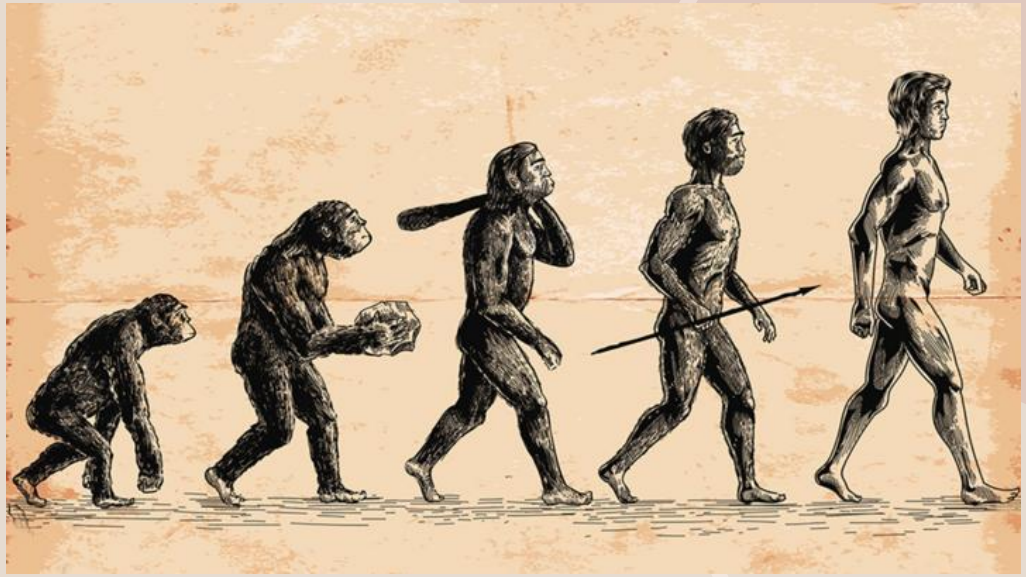
Multiple components must agree on the ORDER OF VALUES, e.g. multiple parameters in method calls - both caller and callee must agree on the semantics of the first, second, etc. parameters

```
function printRentalStatement()() {
  const frequentRenterPoints =  ${totalFrequentRenterPoints(customer, movies)}`);
  const total      = frequentRenterPoints[0];
  const average = frequentRenterPoints[1];
  // ...
}
```

Connascence of Position

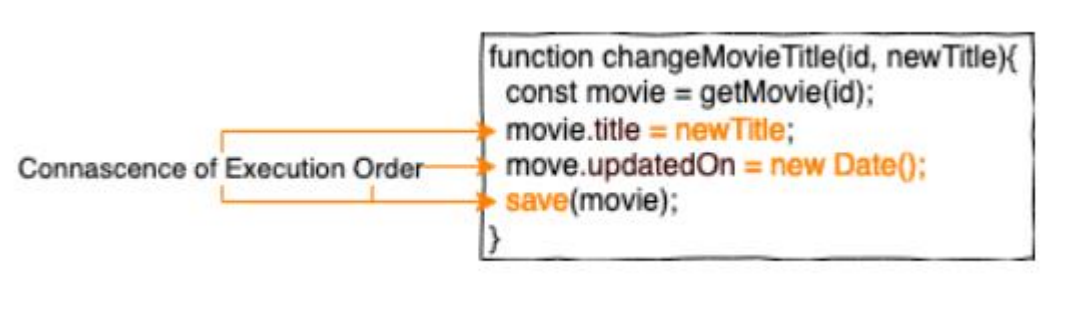
Connascence of Position

```
function frequentRenterPoints(customer, movies)() {
  // ...
  return [4, 12];
}
```



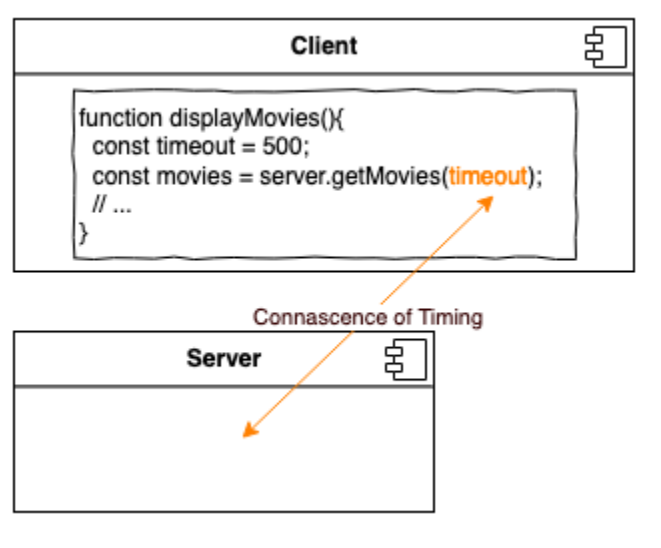
Connascence of Execution Order

Multiple components must executed in a PARTICULAR ORDER, e.g. updating values of an object before saving them



Connascence of Timing

When the timing of the execution of multiple components is important, e.g. client expecting the server to respond after a certain timeout.



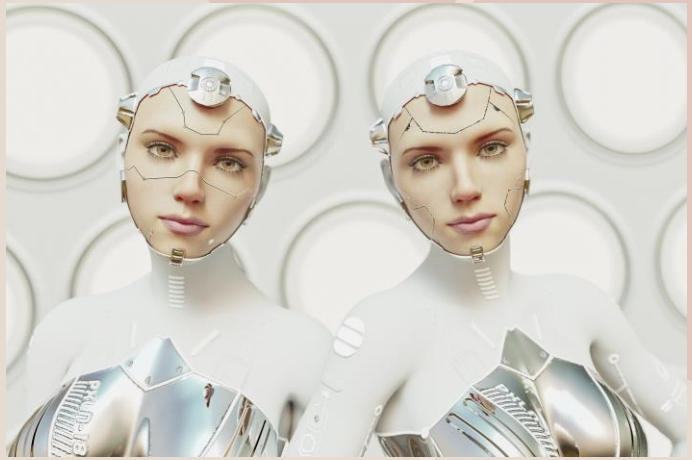
Connascence of Value

when an invariant (permanent condition) states that two or more VALUES must CHANGE SIMULTANEOUSLY.

```
function customerReturnsMovie(movieId, userId){
  // ...
  customerStatistics.moviesRentedByCustomer -= 1;
}

                                        Connascence of Value

function returnMovie(movieId){
  // ...
  globalStatistics.moviesRentedByAllCustomers -= 1;
}
```



Connascence of Identity

Multiple components must REFERENCE the SAME ENTITY

```
function userRequestedTitleChange(movie, newTitle){
  changeMovieTitle(movie.id, newTitle);
  displayMovie(movie);
}

Connascence of Identity

function changeMovieTitle(id, newTitle){
  const movie = getMovie(id);
  movie.title = newTitle;
  move.updatedOn = new Date();
  save(movie);
}
```

Connascence of Manual Task

When manual tasks need to be completed separately OUTSIDE the code as part of the functionality.

STATIC

Can be discovered upon by examining the code

- Name
- Type
- Meaning
- Algorithm
- Position

DYNAMIC

Can be discovered only at runtime

- Execution Order
- Timing
- Value
- Identity
- Manual Task

Benefits of Connascence

- Will help to adhere to many other coding principles – perhaps reducing coupling and maximising cohesion being the biggest beneficiaries
- Helps avoiding code smells e.g. shotgun surgery (though many others)
- Promotes a good design and architecture – refactoring becomes a lot more straightforward, more easily extensible, encourages dependency flows in the right direction

Closing thoughts....

Thanks for listening !

References:

https://www.maibornwolff.de/en/know-how/connascence-rules-good-software-design/
Richard Gross, 2023

Agile Technical Practices Distilled
Alessandro Di Giola, Marco Consolaro, Pedro Moreira Santos, 2019

Wikipedia