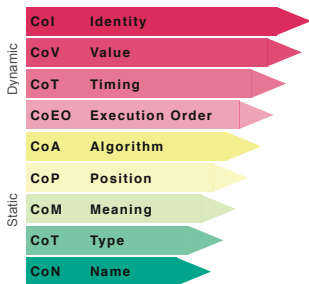
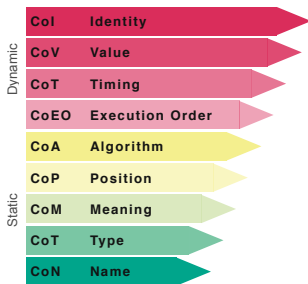


Connascence types



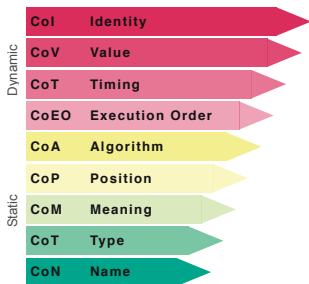
ambientia

Connascence types



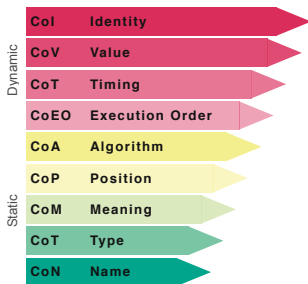
ambientia

Connascence types



ambientia

Connascence types



ambientia

Connascence explained

Connascence is a software quality metric & a taxonomy for different types of coupling.

Dynamic connascence

CoI: Multiple components reference the same entity

CoV: Several values must change together.

e.g. Test knows state of production code

CoT: timing of the execution of multiple components is important

CoE: Order of execution of multiple components is important

Static connascence

CoA: Multiple components must agree on particular algorithm

e.g. Test and production code

CoP: Multiple entities must agree on the order of values, e.g. method parameter

CoM: Multiple components must agree on meaning of particular values

CoT: Agree on the type of entity, e.g. return type

CoN: Agree on a name of entity, e.g. class name

ambientia

Connascence explained

Connascence is a software quality metric & a taxonomy for different types of coupling.

Dynamic connascence

CoI: Multiple components reference the same entity

CoV: Several values must change together.

e.g. Test knows state of production code

CoT: timing of the execution of multiple components is important

CoE: Order of execution of multiple components is important

Static connascence

CoA: Multiple components must agree on particular algorithm

e.g. Test and production code

CoP: Multiple entities must agree on the order of values, e.g. method parameter

CoM: Multiple components must agree on meaning of particular values

CoT: Agree on the type of entity, e.g. return type

CoN: Agree on a name of entity, e.g. class name

ambientia

Connascence explained

Connascence is a software quality metric & a taxonomy for different types of coupling.

Dynamic connascence

CoI: Multiple components reference the same entity

CoV: Several values must change together.

e.g. Test knows state of production code

CoT: timing of the execution of multiple components is important

CoE: Order of execution of multiple components is important

Static connascence

CoA: Multiple components must agree on particular algorithm

e.g. Test and production code

CoP: Multiple entities must agree on the order of values, e.g. method parameter

CoM: Multiple components must agree on meaning of particular values

CoT: Agree on the type of entity, e.g. return type

CoN: Agree on a name of entity, e.g. class name

ambientia

Connascence explained

Connascence is a software quality metric & a taxonomy for different types of coupling.

Dynamic connascence

CoI: Multiple components reference the same entity

CoV: Several values must change together.

e.g. Test knows state of production code

CoT: timing of the execution of multiple components is important

CoE: Order of execution of multiple components is important

Static connascence

CoA: Multiple components must agree on particular algorithm

e.g. Test and production code

CoP: Multiple entities must agree on the order of values, e.g. method parameter

CoM: Multiple components must agree on meaning of particular values

CoT: Agree on the type of entity, e.g. return type

CoN: Agree on a name of entity, e.g. class name

ambientia