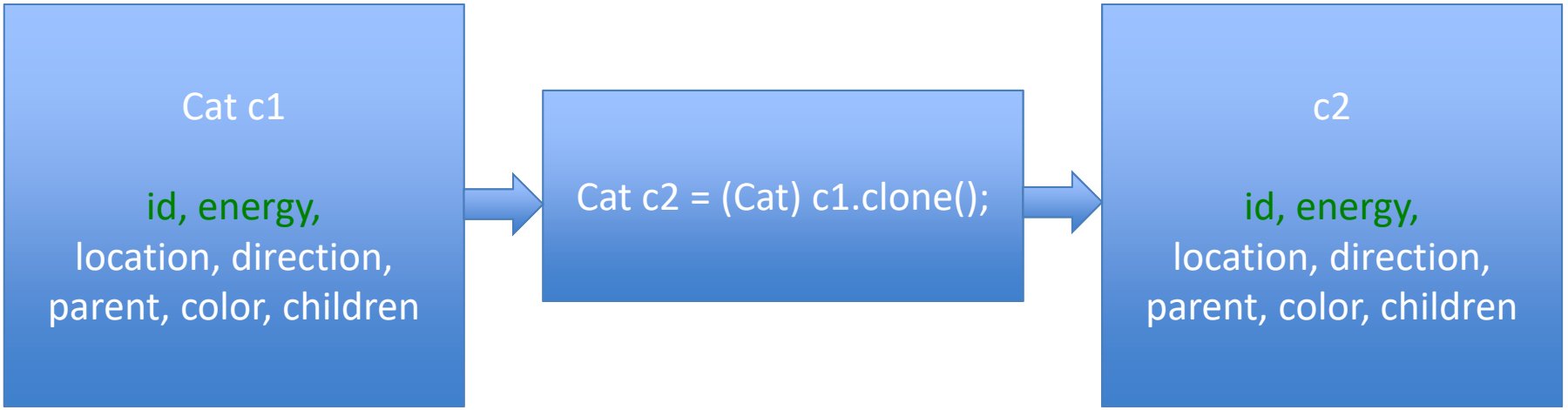
CLONING

Clone/Copy

- exact field-to-field independent copy of an object
- most work is done by the Object class
  - clone of Object class provides “shallow copy”
    - basic type variables are fine
    - object type variables are shared (hence, shallow copy)
  - String is not cloneable; manual copy is needed
  - *final* variables are not copied

Shallow Copy



c1 and c2 each have copies of id and energy, and can independently update them

all other variables are object types and they are shared between them

Cloneable

- Lets Java know this class can be cloned

Steps to Enable Cloning for Class A

- class must implement *Cloneable* (java.lang.Cloneable) interface
- override: *public Object clone()*
- add *try-catch* block to clone() method
- catch *CloneNotSupportedException* and return null

Steps to Enable Cloning

- add following code to the try section of clone() method

```
// returns a cloned object of this (current) A object
// all basic type variables are copied already
A copy = (A) super.clone();

// for all object type variables,
// manually assign this A object's values to copy
copy….
```

Toy Example

```
class Login {
    private String uid;
    private String password;
}
```

Example: Breeding Trap

- Trap from hw3 ~ 5, and exam 1
  - stationary once created
  - breeds all around at death
  - a Trap clone starts with no children to avoid having children change their parent pointer

Example: MyGrid

- MyGrid in hw 4
  - to save current state before a change in grid (can support undo/redo)
  - must deep-clone