ECS: COMPONENTS

```
struct Vel(f32);

impl Component for Vel {
    type Storage = VecStorage<Self>;
}


#[derive(Component, Debug)]

#[storage(DenseVecStorage)]

struct Pos(f32, f32, f32);
```

ECS: SYSTEMS

```
struct TransformSystem;

impl<'a> System<'a> for TransformSystem {
    type SystemData = (WriteStorage<'a, Pos>, ReadStorage<'a, Vel>);


    fn run(&mut self, (mut pos, vel): Self::SystemData) {
        // The `.join()` combines multiple components,
        // so we only access those entities which have
        // both of them.
        for (pos, vel) in (&mut pos, &vel).join() {
            pos.0 += vel.0;
        }
    }
}
```