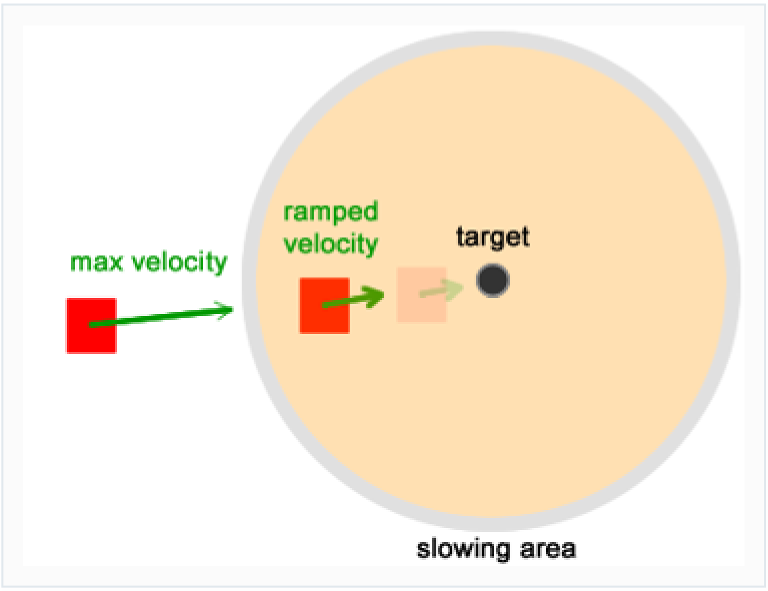
Arrive

The seek behavior, as we've seen, makes a character move towards a target. When it reaches the destination the steering force keeps acting on it based on the same rules, making the character "bounce" back and forth around the target.

The seek behavior. Move the mouse to move the target.
The *arrival* behavior prevents the character from moving through the target. It makes the character slow down as it approaches the destination, eventually stopping at the target.

The behavior is composed of two phases. The first phase is when the character is far away from the target and it works exactly the same way as the seek behavior does (move at full speed towards the target).

The second phase is when the character is close to the target, inside the "slowing area" (a circle centered at the target's position):



When the character enters the circle, it slows down until it stops at the target.

Slowing Down

When the character enters the slowing area its velocity is ramped down linearly to zero. This is achieved by adding a new steering force (the *arrival force*) to the character's velocity vector. The result of this addition will eventually become zero, meaning that nothing will be added to the character's position every frame (so there will be no movement):

```
// If (velocity + steering) equals zero, then there is no movement
velocity = truncate(velocity + steering, max_speed)
position = position + velocity

function truncate(vector:Vector3D, max:Number) :void {
    var i :Number;
    i = max / vector.length;
    i = i < 1.0 ? i : 1.0;
    vector.scaleBy(i);
}
```

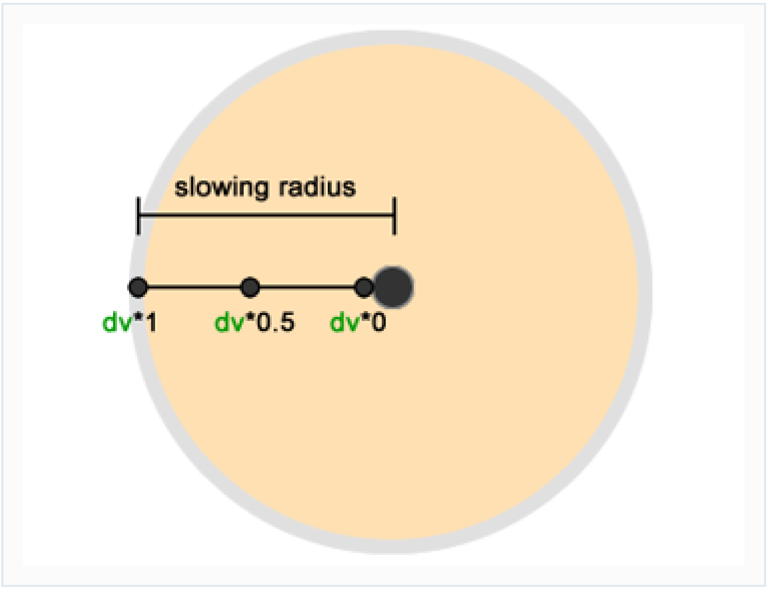
In order to ensure the character will gradually slow down before it stops, the velocity should not become zero immediately. The gradual slow down process is calculated based on the radius of the slowing area and the distance between the character and the target:

```
// Calculate the desired velocity
desired_velocity = target - position
distance = length(desired_velocity)

// Check the distance to detect whether the character
// is inside the slowing area
if (distance < slowingRadius) {
    // Inside the slowing area
    desired_velocity = normalize(desired_velocity) * max_velocity * (distance / s
} else {
    // Outside the slowing area.
    desired_velocity = normalize(desired_velocity) * max_velocity
}

// Set the steering based on this
steering = desired_velocity - velocity
```

If the distance is greater than `slowingRadius`, it means the character is far away from the target and its velocity should remain `max_velocity`.

If the distance is less than `slowingRadius`, it means the character has entered the slowing area and its velocity should be ramped down.

The term `distance / slowingRadius` will vary from `1` (when `distance` equals `slowingRadius`) to `0` (when `distance` is almost zero). This linear variation will make the velocity smoothly slow down:



As previously described, the character movement is performed as follows:

```
steering = desired_velocity - velocity
velocity = truncate (velocity + steering , max_speed)
position = position + velocity
```

If the desired velocity is ramped down to zero, then the steering force becomes `-velocity`. As a consequence, when that steering force is added to the velocity, it will result in zero, making the character stop moving.

Below is a demo showing the arrival behavior:

Move the mouse to move the target.
What the arrival behavior does, really, is calculate a force that will be equal to `-velocity`, preventing the character from moving as long as that force is in place. The character's original velocity vector does not change and it continues to work, but it is nulled by the steering addition.

If the arrival steering force is lifted, the character will start moving again, using its original velocity vector.