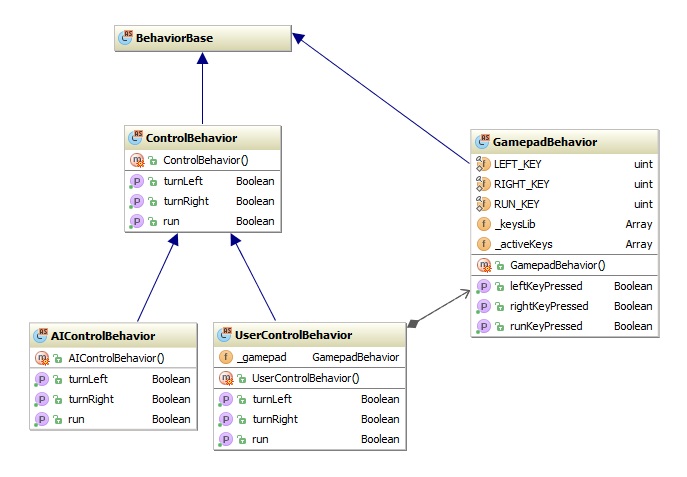
В первой части был описан пример реализации основ концепции поведений. Теперь конкретнее.

Одна из наиболее частых задач, возникающих при разработке игры – организация управления объектами игрового мира. Первое, что приходит в голову – передвижение по полю.

Итак, поведение передвижения:



Шаг основного цикла работы doStep() проверяет, содержит ли контроллер объекта поведение ControlBehavior – является ли он управляемым. И, в зависимости от триггеров run, turnRight и turnLeft, применяет соответствующие действия к игровой сущности.

Что же такое ControlBehavior:

ControlBehavior является абстрактным классом, описывающим набор триггеров для потомков. Более смысловой нагрузки он не несет, и сам по себе не используется. Если мы хотим получить бот, то используем AIControlBehavior. Если нам нужен объект, управляемый пользователем добавляем в его контроллер UserControlBehavior. Остановимся на последнем подробнее.

Поведение, контролируемое пользователем агрегирует GamepadBehavior, которое отслеживает нажатые клавиши. Таким образом, тот факт, стоит ли ObjectBase бежать, поворачивать налево или на право – зависит от нажатой клавиши. Что верно только для UserControlBehavior.



Таким образом, это всего лишь «обертка» над клавиатурой. А что будет активировать триггеры AIControlBehavior – заставлять ехать и поворачивать бот – знать будет только AI. Например, в одном из моих текущих проектов он меняет направление движения в зависимости от векторов между соседними вейпоинтами.

Вернемся к поведению передвижения. Использование при выборе дальнейших действий игрового объекта ControlBehavior позволяет внутри MoveBehavior не думать о том, кто управляет. Это ведь поведение передвижения - не более.

Собственно на этом все. В конце хотелось бы отметить, что я использую данный подход - концепцию поведений - в приведенном здесь виде. Но это совершенно не значит, что он подойдет всем вам в любом проекте.

Надеюсь, что прочитанное направит ваши мысли в нужное русло и так или иначе поможет реализовать весь требуемый функционал. При этом уложиться в сроки и создать максимально качественный продукт. Удачи в разработке игр!