Kodu Language and Grammar Specification

Kathryn T. Stolee

August 27, 2010

We describe the language of Kodu using a grammar based on the notation for context-free grammars. This language specification should serve as a reference for researchers and teachers who seek to learn or study Kodu as a language. To make these resources more accessible to a broader audience, we have generated two different language descriptions. The first is a basic language description, which provides the general structure and syntax of a Kodu program. The second is an extension of the basic language that contains all the constructs implemented in the Kodu language.

1 Basic Kodu Language

Kodu is a high-level, visual, and interpreted language. It is heavily inspired by robotics, and as such, each character and object in Kodu is programmed individually to interact with the world, much like intelligent agents.

1.1 About the Language

The Kodu language is entirely event driven, where each line of programming is in the form of a condition and an action, referred to as a *rule* (different from the production rules used in the language description). For example, a rule could read, *when see apple red, do move toward quickly*, where *when see apple red* is the conditional, and *do move toward quickly* is the action. Each word in the rule (omitting *when* and *do*), is represented as a tile in Kodu, and thus is a member of the alphabet.

This language description is represented by a series of *production rules*, where the left-hand side (LHS) shows a *variable*, also known as a *non-terminal*, and the right-hand side (RHS) contains variables and *terminals*. Each terminal is an element on the *alphabet* of the Kodu language and all begin with a lower-case letter. In the case of Kodu, the alphabet is composed of the entire set of *tiles* available during programming.

As an example of how to read a grammar and production rules, we generate a grammar that can represent an example rule, *when see apple do move toward quickly*. This is shown in Figure 1.1 (recall that *when* and *do* are not part of the alphabet, and so they are not in the grammar). Here, the alphabet $\Sigma = \{$ see,

apple, do, move, toward, quickly}, the set of variables $V =$ { Rule, Condition, Action, Sensor, Filter, Actuator, Selector, Modifier}, and the start variable $S \in V$ is *Rule*.

Rule	$\rightarrow$	Condition Action
Condition	$\rightarrow$	Sensor FilterSet
Action	$\rightarrow$	Actuator Modifier Selector
Sensor	$\rightarrow$	see
FilterSet	$\rightarrow$	Filter FilterSet | Filter
Filter	$\rightarrow$	apple | red
Actuator	$\rightarrow$	move
Selector	$\rightarrow$	toward
Modifier	$\rightarrow$	quickly

Figure 1: Simple Grammar

From the simple grammar definition in Figure 1.1, we can now derive the example rule, and show this derivation in Figure 1.1. We begin with the start variable, *Rule*, and use substitution to arrive at the example rule, *see apple red, move toward quickly*.

Rule	$\Rightarrow$	Condition, Action
	$\Rightarrow$	Sensor FilterSet, Action
	$\Rightarrow$	see FilterSet, Action
	$\Rightarrow$	see Filter FilterSet, Action
	$\Rightarrow$	see apple FilterSet, Action
	$\Rightarrow$	see apple Filter, Action
	$\Rightarrow$	see apple red, Action
	$\Rightarrow$	see apple red, Actuator Selector Modifier
	$\Rightarrow$	see apple red, move Selector Modifier
	$\Rightarrow$	see apple red, move toward Modifier
	$\Rightarrow$	see apple red, move toward quickly

Figure 2: Derivation of Example Rule using Simple Grammar

1.2 Kodu Basic Grammar

The grammar given in Figure 1.1 is a simplified grammar given for a single rule in a Kodu program, but each Kodu program has many rules and as well as additional constructs that define the organization of those rules.

Figure 3 shows the basic grammar for the Kodu language, and is an extension of the simple grammar described previously. All non-terminals begin with an upper-case letter, and all terminals are lower-cased. Most of the terminals are not listed for brevity. *Game* is the start variable, and it has a variable *Actors*, which is a set of *Objects*. Each Object's programming is defined by at least one *Page*, and each Page has one or more *Rules*. A Rule is defined as a *Condition*

Action, which may or may not be followed by nested rules (represented by the *Page* on the RHS of the *Rule* production). Each rule is broken into a *Condition Action* sequence, as described previously.

Game	$\rightarrow$	Actors
Actors	$\rightarrow$	Object | Object Actors
Object	$\rightarrow$	Page Object | Page
Page	$\rightarrow$	Rule Page | Rule
Rule	$\rightarrow$	Condition Action | Condition Action Page
Condition	$\rightarrow$	Sensor FilterSet | ϵ
Action	$\rightarrow$	Actuator Selector ModifierSet | Actuator ModifierSet | ϵ
ModifierSet	$\rightarrow$	Modifier ModifierSet | Modifier
FilterSet	$\rightarrow$	Filter FilterSet | Filter
Sensor	$\rightarrow$	see | hear | bump | ...
Filter	$\rightarrow$	apple | blue | health | ... | ϵ
Actuator	$\rightarrow$	move | shoot | add | ...
Selector	$\rightarrow$	toward | me | avoid | ... | ϵ
Modifier	$\rightarrow$	5 points | red | quickly | ... | ϵ

Figure 3: Basic Kodu Grammar

Using the example rule, *when see apple red, do move toward quickly*, we derive the syntax using this grammar, shown in Figure 4. This time, the derivation is given using a parse tree instead of rule substitution. Each rectangle represents a non-terminal in the grammar, and the quoted words represent terminals.

1.3 Full Kodu Grammar

The Kodu language currently consists of over 500 tiles with a general structure that mimics that shown in Figure 3. As the grammar involves over 130 non-terminals (and thus the same number of production rules), we show this grammar in Appendix 2.

The implementation of the Kodu grammar within the Kodu Game Lab imposes some restrictions on the grammar as defined (e.g., *PageNumber* has a range from 1 – 12), and relaxes the RHS of some rules (e.g., in *DoScoring* the *once* terminal can appear anywhere after the *ScoreTiles* non-terminal). We do not list these constraints and relaxations explicitly, but the impact is that the user has more freedom in the ordering of the tiles. For example, the tile sequence *score 100 red once* is semantically equivalent to *score once red 100*.

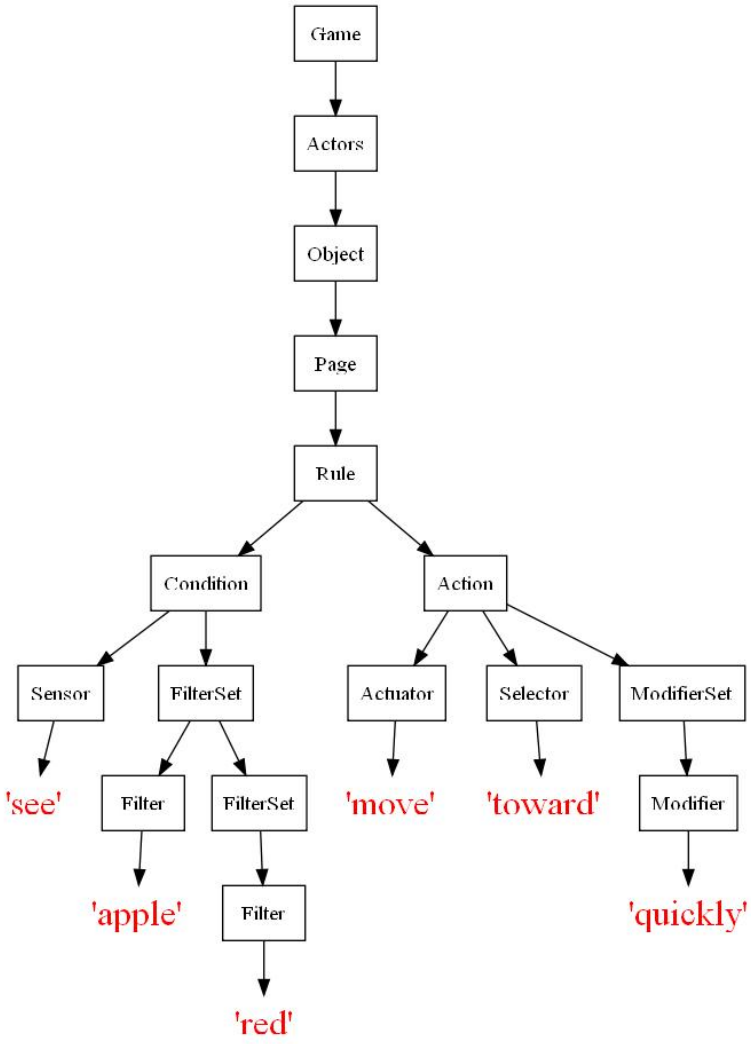

Figure 4: Derivation of *when see apple red, do move toward quickly* using basic grammar

2 Full Grammar Sketch

Some of the non-terminals are appended with the letters, *DO*. This indicates the case when the condition results in a direct object on which the action execute. For example, *when see red apple* creates a direct object, the red apple, and you can *move toward* it. On the other hand, *when gamepad A* does not create a direct object, so you cannot move toward anything.

Game	→	GameActors | ϵ
GameActors	→	Actor GameActors | Actor
Actor	→	Page Actor | Page
Page	→	Rule Page | Rule | ϵ
Rule	→	ConditionAction | ConditionAction Page | ConditionAction | ConditionAction Page | MeAction | MeAction Page
ConditionAction	→	Conditions not | Conditions
Conditions	→	DOConditions DOAction | OtherConditions MeAction | HeldByAction
DOConditions	→	WhenMouseDO | WhenSight | WhenHearing | WhenBump | WhenShotHit
OtherConditions	→	WhenGamePad | WhenKeyBoard | WhenMouseOther | WhenTimer | WhenGot | WhenScore | WhenHealth | WhenOnLand | WhenOnWater | WhenAlways
DOAction	→	DoMovementDO | DoTurningDO | DoEatDO | DoLaunchDO | ActuatorsDirectObject | DoHoldingDO | DoResetDO | GenericAction
MeAction	→	DoMovement | DoTurning | DoEatOther | DoLaunch | ActuatorsMeObject | DoHolding | DoReset | GenericAction
GenericAction	→	DoSay | DoOpenClose | DoCreate | DoSound | DoJump | DoSwitch | DoShoot | DoEndGame | DoScoring | DoCamera
HeldByAction	→	DoTurning | DoEatDO | DoLaunchDO | ActuatorsDirectObject | DoResetDO | GenericAction
ActuatorsDirectObject	→	ActuatorsVariableObject DirectObjectModifier
ActuatorsMeObject	→	ActuatorsVariableObject MeModifier
ActuatorsVariableObject	→	Remove | DamageHeal | GlowColorExpress
DoCamera	→	follow | ignore | firstperson
DoScoring	→	ScoreTiles NumberComparisonFilter OnceModifier
ScoreTiles	→	score | unscore
OnceModifier	→	once | ϵ
DoEndGame	→	end | victory PlayerFilter ColorFilter
DoReset	→	ResetActuator HealthGlowExpress MeModifier | ResetWorld
DoResetDO	→	ResetActuator HealthGlowExpress DirectObjectModifier | ResetWorld

ResetActuator	→	reset
ResetWorld	→	ResetActuator WorldScoreModifier
MeModifier	→	me | ϵ
HealthGlowExpress	→	ResetHealthModifier ResetGlowModifier ResetExpressModifier OnceModifier
ResetHealthModifier	→	health | ϵ
ResetGlowModifier	→	glow | ϵ
ResetExpressModifier	→	express | ϵ
WorldScoreModifier	→	world ScoreBucketFilter | score ScoreBucketFilter OnceFilter | world score ScoreBucketFilter
DoHolding	→	grab OnceFilter | give | drop
DoHoldingDO	→	grab OnceFilter ItModifier | give | drop
Remove	→	CombatModifiers OnceModifier
CombatModifiers	→	vanish | boom | knockout | stun
DamageHeal	→	DamageOrHeal ScoreFilter RandomFilter OnceModifier
DamageOrHeal	→	damage | heal
BlipMissileModifier	→	blip | missile | ϵ
CardinalDirection	→	NSModifier EWModifier
NSModifier	→	north | south | ϵ
EWModifier	→	east | west | ϵ
UpDownModifier	→	up | down | ϵ
DoShoot	→	shoot BlipMissileModifiers OnceModifier
BlipMissileModifiers	→	MissileOrBlip BlipMissileOptions
MissileOrBlip	→	Blip | Missile
Blip	→	blip | ϵ
Missile	→	missile LevelCruise | ϵ
BlipMissileOptions	→	DirectionModifiers ColorFilters CombatOrNone OnceModifier
CombatOrNone	→	CombatModifiers | ϵ
LevelCruise	→	level | cruise | ϵ
DirectionModifiers	→	CardinalUpDown | forward
CardinalUpDown	→	CardinalDirection | UpDownModifier | ϵ
DoSwitch	→	switch TaskModifier
TaskModifier	→	page PageNumber
PageNumber	→	0 | 1 | 2 | ...
DoJump	→	jump HighLowModifier OnceModifier
HighLowModifier	→	HighModifier | LowModifier
HighModifier	→	high HighModifier | ϵ
LowModifier	→	low LowModifier | ϵ
DoSound	→	QuietOrPlay AnyAllSounds OnceModifier
QuietOrPlay	→	quiet | play
AnyAllSounds	→	anysound | SoundFilter | ϵ
GlowColorExpress	→	DoGlow | DoColor | DoExpress |
DoGlow	→	glow GlowColorsOff OnceModifier
GlowColorsOff	→	ColorFilter | glowoff | ϵ

DoColor	→	color ColorFilter OnceModifier
DoExpress	→	express ExpressionFilter OnceModifier
DoLaunchDO	→	DoLaunch ItModifier
DoLaunch	→	launch ColorFilter ObjectCreatable StrengthModifier CardinalUpDown HighLowModifier OnceModifier
StrengthModifier	→	WeakModifier | StrongModifier
WeakModifier	→	weak WeakModifier | ϵ
StrongModifier	→	strong StrongModifier | ϵ
DoCreate	→	create ColorFilter ObjectCreatable OnceModifier
ObjectCreatable	→	ObjectModifier | CreatableModifier | ϵ
ObjectModifier	→	rock | apple | star | coin | heart | ball | ammo
CreatableModifier	→	creatable
OnceMeModifier	→	OnceModifier MeModifier
DirectObjectModifier	→	MeModifier | ItModifier
ItModifier	→	it | ϵ
DoOpenClose	→	OpenClose OnceMeModifier
OpenClose	→	open | close
DoSay	→	text OnceModifier
DoEatDO	→	DoEat DirectObjectModifier
DoEat	→	eat OnceModifier
DoTurning	→	turn TurnDirection SpeedModifier
DoTurningDO	→	turn TurnDirectionDO SpeedModifier
TurnDirection	→	forward | left | right | ϵ
TurnDirectionDO	→	toward | TurnDirection
SpeedModifier	→	SlowModifier | FastModifier
SlowModifier	→	slowly SlowModifier | ϵ
FastModifier	→	quickly FastModifier | ϵ
DoMovement	→	move MovementModifiers ConstraintModifiers SpeedModifier
DoMovementDO	→	move MovementModifiersDO ConstraintModifiers SpeedModifier
MovementModifiers	→	CardinalDirection | wander | forward | followpath ColorFilter | ϵ
MovementModifiersDO	→	toward | away | avoid | circle RightLeftFilter RangeFilter | MovementModifiers
ConstraintModifiers	→	NSEWConstraints | freeze | ϵ
NSEWConstraints	→	NSConstraintModifier EWConstraintModifier
NSConstraintModifier	→	ns | ϵ
EWConstraintModifier	→	ew | ϵ
WhenGamePad	→	gamepad GamePadFilter PlayerFilter
GamePadFilter	→	GamePadStickFilter | GamePadButtonFilter
GamePadStickFilter	→	GamePadSticks DirectionFilter
GamePadSticks	→	lstick | rstick
GamePadButtonFilter	→	abutton | bbutton | xbutton | ybutton | ltrigger | rtrigger | ϵ
PlayerFilter	→	player1 | player2 | player3 | player4 | ϵ

DirectionFilter	→	UpDownFilter RightLeftFilter
UpDownFilter	→	DirectionUpFilter | DirectionDownFilter | ϵ
RightLeftFilter	→	DirectionRightFilter | DirectionLeftFilter | ϵ
DirectionUpFilter	→	up
DirectionDownFilter	→	down
DirectionRightFilter	→	right
DirectionLeftFilter	→	left
WhenKeyBoard	→	keyboard KeyBoardKeyFilter
KeyBoardKeyFilter	→	akey | bkey | ... | zkey | d0key | d1key | ... | d9key | f1key | f2key | ... | f12key | spacekey | pageupkey | ...
WhenMouseDO	→	mouse MouseSelect ExplicitSubjectTerrain
WhenMouseOther	→	mouse MouseMove
MouseMove	→	move
MouseSelect	→	leftbutton | rightbutton | hover
ExplicitSubjectTerrain	→	TerrainFilter | ExplicitSubject
ExplicitSubject	→	ObjectFilter DescriptionFilter MeFilter
ObjectFilter	→	kodu | anything | flyfish | jet | light | cycle | saucer | blimp | balloon | sub | cannon | puck | wisp | anybot | turtle | pushpad | sputnik | stick | drum | mine | cloud | fish | ship | factory | hut | castle | tree | anybuilding | ObjectModifier | ϵ
MeFilter	→	me | ϵ
DescriptionFilter	→	ColorFilter ExpressionFilter RangeFilter
ColorFilter	→	black | grey | white | red | orange | yellow | green | blue | purple | pink | brown | ϵ
ExpressionFilter	→	happy | sad | angry | crazy | hearts | flowers | stars | swears | blank | ϵ
RangeFilter	→	CloseFilter | FarFilter
CloseFilter	→	close CloseFilter | ϵ
FarFilter	→	far FarFilter | ϵ
WhenSight	→	see ExplicitSubject
WhenHearing	→	hear ExplicitSubjectSounds
ExplicitSubjectSounds	→	SoundFilter ObjectFilter DescriptionFilter MeFilter
SoundFilter	→	... | ϵ
WhenBump	→	bump ExplicitSubjectNoRange
ExplicitSubjectNoRange	→	ObjectFilter ColorFilter ExpressionFilter MeFilter
WhenTimer	→	timer TimerFilter RandomFilter
TimerFilter	→	Times TimerFilter | Times
Times	→	0.25s | 1s | 2s | 3s | 4s | 5s | 10s | 20s | 30s | 60s
RandomFilter	→	random | ϵ
WhenGot	→	got ExplicitSubjectNoRange
WhenScore	→	scored NumberComparisonFilter ScoreBucketFilter
ScoreFilter	→	Scores | Scores ScoreFilter
Scores	→	0 | 1 | 2 | 3 | 4 | 5 | 10 | 20 | 50 | 100

ScoreBucketFilter	→	whitebucket | blackbucket | greybucket | redbucket | greenbucket | bluebucket | orangebucket | yellowbucket | purplebucket | pinkbucket | brownbucket | abucket | bbucket | ... | zbucket
ScoreCompareFilter	→	scoreis | scoreabove | scorebelow
WhenHealth	→	health NumberComparisonFilter
NumberComparisonFilter	→	ScoreFilter RandomFilter ScoreBucketFilter
WhenShotHit	→	hit ExplicitSubject
WhenHeldBy	→	held ExplicitSubjectNoRange
WhenOnLand	→	terrain TerrainFilter
TerrainFilter	→	... | ϵ
WhenOnWater	→	water WaterFilter
WaterFilter	→	... | ϵ
WhenAlways	→	always