A Programming Language for Future Interests

James Grimmelmann

University of Cincinnati College of Law
Corporate Law Symposium
March 12, 2021
In this talk

• We made a thing
• How it works
• Why it matters
Collaborators

- Shrutarshi Basu (Harvard CS)
- Nate Foster (Cornell CS)
- Shan Parikh (Cornell ’21)
- Ryan Richardson (Cornell ’21)
I. We made a thing
Two kinds of Property students

• “Future interests don’t make any sense.”

• “Future interests are the only part of this course that makes any sense.”
A common intuition

- Estates and future interests are different
- Super-bright-line rules
- Highly mechanical
- Rigid syntax
- Like learning a foreign language …
- … or like learning a *programming* language
Demo
What was *that*?

- We taught a computer to how to interpret conveyances of future interests
- Live online at https://conveyanc.es
- *Property Conveyances as a Programming Language* (Onward 2019), and *A Programming Language for Future Interests* (in submission)
II. How it works
Orlando: a language

Littleton: an interpreter
Three big ideas

• *Syntax* for the language of conveyances
• *A data structure* for the state of title
• *Semantics* to update in response to events
Syntax: a formal grammar

grant  ->  to  person  duration
grant  ->  grant, then  grant

person  ->  Alice
person  ->  Bob

duration  ->  for  life
duration  ->  and  pronoun  heirs
Parsing a conveyance

to Alice for life, then to Bob and his heirs
to *person* for life, then to Bob and his heirs
to *person duration*, then to Bob and his heirs
*grant*, then to Bob and his heirs
*grant*, then to *person* and his heirs
*grant*, then to *person* and *pronoun* heirs
*grant*, then to *person duration*
*grant*, then *grant*
*grant*
A data structure: title trees

while Slughorn does not consume chocolate

while Slughorn is alive

Slughorn has a possessory estate in life estate subject to executory limitation

Willy Wonka has a reversion in fee simple subject to executory limitation

Charlie Bucket has an executory interest in fee simple
Translating conveyances

to A

A has a possessory estate in fee simple

for life

while A is alive

to A for life

while A is alive

A has a possessory estate in life estate
Semantics

while A is alive

A has a possessory estate in life estate

B has a remainder in fee simple (vested)

A dies

B has a possessory estate in fee simple
$$
\begin{align*}
\delta(\text{to } p) &= \text{to } p \\
\delta(\bot) &= \bot \\
\delta(t \text{ while } c) &= \begin{cases} 
\delta(t) & \text{if } \vdash c \text{ and } \delta(t) \neq \bot \\
\bot & \text{if } \nvdash c \\
\bot & \text{if } \delta(t) = \bot 
\end{cases} \\
\delta(\text{if } c \text{ then } t_1 \text{ else } t_2) &= \begin{cases} 
\delta(t_1) & \text{if } \vdash c \\
\delta(t_2) & \text{if } \nvdash c 
\end{cases} \\
\delta(t_1 \rightarrow t_2) &= \begin{cases} 
\delta(t_1) \rightarrow t_2 & \text{if } \delta(t_1) \neq \bot \\
\delta(t_2) & \text{if } \delta(t_1) = \bot 
\end{cases}
\end{align*}$$
Modeling property law

• **Quanta:** fee simple, fee tail, life estate, term of years
• Special limitations, conditions precedent, conditions subsequent, executory limitations
• Implied reversions, multiple conveyances
• Naming, vesting, and the Rule Against Perpetuities
III. Why it matters
A mirror of property law

• There’s a reason future interests are so arid
  • The way they’re taught, they’re close to a programming language already
• The grammar of future interests is:
  • Recursive
  • Modular
Programming languages and legal language

• Orlando captures the linguistic structure of property conveyances: a flexible set of basic elements combined according to fixed rules

• Some areas of law are well-suited to this:
  • Contracts
  • Tax
  • Statutory drafting
Applications

• Teaching tool
• Easy visualization
• Highly interactive
• Foundation for future scholarship
• Foundation for future practitioner tools
Discussion