

A Review of Extensional Semantics, Part 1: The Fundamentals of Extensional Semantics ¹

1. The Big Picture

(1) The Ultimate Goal

A precise, formal theory of a particular sub-component the human language faculty:
the ability to productively interpret the (infinite) sentences of our language(s).

(2) The Productivity of Interpretation: Compositionality

Every human language contains a (discretely) infinite number of meaningful structures. How do speakers of a language know the meaning of each of these infinite number of sentences?

a. Memorization of every sentence? Of course not:

- i. The human brain is a finite object.
- ii. Speakers can comprehend sentences they've never heard before
(*Einstein elected French fries.*)

b. The Core Hypothesis (Compositionality)

The meaning of a complex structure is *derived* by a system that consists of (at least) the following:

- i. A finite set of stipulated semantic entries (words, idioms)
- ii. A set of rules by which the meaning of a complex structure can be computed via the meanings of its component parts.

Meaning('Einstein elected French fries') =
Meaning('Einstein') + Meaning('elected') + Meaning('French fries')

(3) The Ultimate Goal, Restated

What we want is a precise, formal characterization of:

- (a) The basic semantic entries for the lexical items of a language
- (b) The compositional rules that derive the meaning of larger structures

And we want this theory to make only correct predictions regarding the meanings that the structures of a language can have.

¹ These notes encapsulate much of the information in Heim & Kratzer (1998), Chapters 1 – 5.

(4) **A Crucial, Foundational Question: “What is a ‘meaning’?”**

What is a ‘meaning’ such that it can combine with other ‘meanings’?

Problem:

The word ‘meaning’ is a vague, pre-theoretic term from every-day discourse. Thus, it may not be an appropriate term for a precise, scientific study of human language. (cf. biology doesn’t actually employ terms like ‘alive’ or ‘dead’ or ‘life-form’)

... so how can we better pin down the phenomena/properties that we are interested in, those that are typically, loosely categorized under the general umbrella of ‘meaning’?...

(5) **‘Meaning’ is as ‘Meaning’ Does**

“In order to say what a meaning is, we may first ask what a meaning *does*, and then find something that does that” (David Lewis; “General Semantics”)

What (exactly) do we *know* when we *know* ‘the meaning’ of a sentence?
Well, we know a lot of things!

(a) Social Appropriateness of the Statement:

What social contexts the statement is appropriate in.

“It’s nice to see you.” vs. “Where you been, man?”

(b) Emotive Content of the Statement:

What the statement reveals about the emotive state of the speaker.

“I disagree with Dave’s judgment.” vs. “Dave is a damn fool!”

(c) Presuppositions of the Statement:

What the statement takes to be granted or already established in the discourse.

“When did you stop beating your wife?” vs. “Did you ever beat your wife?”

(d) **The Informational Content of the Sentence**

What the statement communicates regarding the facts in the world.

Rightly or wrongly, it’s (d) that has received by-far-and-away the greatest attention over the centuries (though (c) has also received a good deal of study in the past thirty years or so).

It will also be the aspect of meaning that we will (almost exclusively) be concerned with in this course. Thus, let’s try to develop our concept of what this ‘informational content’ is...

2. Informational Content as ‘Truth Conditions’

(6) Fundamental Assumption of the Field

The ‘informational content’ of a sentence can be identified as the sentence’s ‘truth conditions’, the *conditions which must hold in order for the sentence to be true*.

What motivates the claim in (6)? Well, let’s first consider a crude model of communication...

(7) A Crude Model of Communication

- (a) Speaker utters a string of sounds (/ ðə haʊs ɪz ən fɑɪ /)
- (b) Listener hears those sounds.
- (c) Listener’s linguistic system interprets the sounds and puts a ‘picture’ (‘conceptual structure’) in their head.



This model is overly simplistic on a number of grounds. Comprehension of a sentence is much more than a little ‘picture’ appearing in your head... Here’s what might be taken as a more sophisticated characterization of the process:

(8) A More Complete Model of Communication

Listener’s cognitive system performs the following computations:

Syntax: / ðə haʊs ɪz ən fɑɪ / has the structure [[the house][is[on[fire]]]]

Semantics: The structure ‘[[the house][is[on[fire]]]]’ is true *iff* the house is on fire.

Pragmatics: The speaker is an honest guy, so he must believe what he is saying...

The speaker is a smart guy, so most things he believes are true...

So “[[the house][is[on[fire]]]]” must be true...

So, (given the output of semantics), the house is on fire...

OH MY GOD THE HOUSE IS ON FIRE!!!

3. Getting Truth Conditions from ‘Extensions’

We ended our ‘interim summary’ in (11) with the following crude sketch of our desideratum:

(12) Crude Sketch of a Compositional Semantic Theory that Yields Truth Conditions

SEMANTICS(‘Dave’) + SEMANTICS(‘smoke’) =
SEMANTICS(‘Dave smokes’) =
TRUTH-CONDITIONS(‘Dave smokes’)

- How do we construct a theory that obtains this result for us?
- What are the ‘meanings’ of “Dave” and “smokes” such that combining them together can yield for us the truth conditions of “Dave smokes”?

... more in a moment!

3.1 More about the Meaning of ‘Meaning’: An Excursus of ‘Extensions’

We’ve already seen that the everyday word ‘meaning’ is vague and confusing in a number of ways...*here’s another*:

(13) The Meaning of the Phrase ‘The President’

- a. In one sense, the *meaning* of the phrase “the president of the United States” has just recently changed. It’s gone from *meaning* George Bush (with all of his attendant connotations) to *meaning* Barack Obama.
(*cf.* ‘denotation’, ‘reference’)
- b. In another sense, the *meaning* of the phrase has stayed the same. It still *means* ‘the person who holds the office of the presidency of the United States’.
(*cf.* ‘sense’, ‘concept’)

Instead of using the word ‘meaning’ in this vague and ambiguous fashion, let’s introduce two different terms to refer unambiguously to these two different ‘senses’ of the word “means”.

(14) Extension vs. Intension

- a. The *extension* of a phrase is the thing in the actual world that the phrase refers to.
 - The *extension* of “the president” is *Barack Obama*.
- b. The *intension* of a phrase is the ‘general concept’ behind the phrase, which determines (for a given world/situation) what the extension of the phrase is.
 - The *intension* of “the president” is *the person who holds the office...*

...So, using this terminology, it's clear the *meaning* of a definite DPs like 'the president' can be broken down (in part) into the phrase's 'extension' and its 'intension'...

...But what about the 'meaning' of a sentence? Does this consist of an 'extension' as well as an 'intension'?

(15) **Intension of a Sentence = 'Truth Conditions'**

Under our new terminology, we might take the 'intension' of a sentence to be its *truth conditions*...

As we've already seen, the truth conditions of a sentence are akin to what we might vaguely describe as the sentence's 'conceptual/propositional content'.

... But if the 'intension' of a sentence is its truth conditions, what is its 'extension'?

(16) **Extension of a Sentence = Truth Value**

If we take the 'intension' of a sentence to be its truth conditions, then we should take the 'extension' of a sentence to be its *truth value*.

Why?

Recall that the 'intension' determines for a given/world situation what the *extension* is. *Truth conditions* determine for a given/world situation what the *truth value* is.

(17) **The General Picture**

a. Intension of "X":

A kind of 'concept' / 'formula' / 'function' which – for any given world/situation – determines what "X" 'picks out' in that world/situation.

b. Extension of "X"

The thing which, in a given world/situation "X" 'picks out'

c. Illustrative Paradigm:

i. Intension of "the president" = *whoever holds the office of the presidency*

ii. Extension of "the president" = Barack Obama

iii. Intension of "the president smokes" = "t.p.s" is T iff the president smokes

iv. Extension of "the president smokes" = TRUE

What's the point of all of this?

You can actually build a decent theory of how meanings can 'compose' to yield truth-conditions by paying attention to extensions (defined as above)!

3.2 Towards a Compositional Semantics Based on Extensions

(18) The Compositionality of Extensions

- a. We've broken down the meaning of a phrase into its *extension* and its *intension*.
- b. Recall, however, that our semantic system must be such that the 'meaning' of a complex expression should be *derived from* the meaning of its component parts.
- c. *Thus, the extension of a complex expression should be derived from the extensions of its component parts!*
- d. A Picture:
 $\text{EXTENSION}(\text{"Obama"}) + \text{EXTENSION}(\text{"smokes"}) = \text{EXTENSION}(\text{"Obama smokes"})$

(19) The Extension of a Predicate = Function

- a. In order to make the picture in (18d) work, the extension of the predicate "smokes" must be such that 'combining' it with the extension of "Obama" yields the extension of "Obama smokes".
- b. *How can we model this?* Well, we know the following:
 - (i) $\text{EXTENSION}(\text{"Obama"}) = \text{Obama}$
 - (ii) $\text{EXTENSION}(\text{"Obama smokes"}) = \text{T}(\text{true})$
- c. *So, the extension of "smokes" combines with Obama and produces the value T...*
- d. The Core Idea:
 $\text{EXTENSION}(\text{"smokes"}) =$
A function from entities to truth values, which yields T iff that entity smokes.

(20) Some New Notation

- a. $[[X]]$ = The extension of X
- b. $\lambda x. x \text{ smokes}$ =
The function from entities to truth values, which yields T *iff* that entity smokes.

(21) Interim Summary

$[[\text{Obama}]]$ = Obama $[[\text{smokes}]]$ = $\lambda x. x \text{ smokes}$ $[[\text{Obama smokes}]]$ = T

We're almost there! We have a statement of the extensions for the lexical items "Obama" and "smokes", and we have an idea as to how those extensions 'compose' together...

All we lack is an explicit statement of the rule:

(22) **The Final Step: The Rule of Function Application**

If X is a structure consisting of two daughters – Y and Z – and if $[[Y]]$ is a function whose domain contains $[[Z]]$, then $[[X]] = [[Y]]([[Z]])$ [i.e., the extension of X is equal to the extension of Y taking as argument the extension of Z].

With the rule in (22), we now have a system that derives the extension of the sentence "Obama smokes" from the extension of its component pieces! *Here's an illustrative derivation:*

(23) **Computing the Extension of 'Obama smokes'**

- a. $[[\text{Obama smokes}]]$ = (by Rule (22))
- b. $[[\text{smokes}]]([[\text{Obama}]])$ = (by definitions in (21))
- c. $[\lambda x. x \text{ smokes}](\text{Obama})$ = (by definition in (20b) and the facts of the world)
- d. T

3.3 From Extensions to Truth Conditions

OK, so we have a system that can compute the *extensions* of complex phrases from the extensions of their component parts...

... but still, *so what?*

How does any of this advance our goal of having a system that derives truth conditions??

Well, let's recall what we want:

(24) **Our Desired Semantic System**

A precise, formal system that for every sentence S of the language, will derive a *correct* statement of the following form: " S is True iff X "

Believe it or not, our compositional semantics for extensions achieves what we want in (24)!!!

(25) **Deriving the Truth Conditions of a Sentence via the Function “[[.]]”**

To Prove: “Obama smokes” is T iff Obama smokes

- a. “Obama smokes” is T *iff* (by definition of our notation)
- b. $[[\text{Obama smokes}]]$ = T *iff* (by Rule (22))
- c. $[[\text{smokes}]]([\text{Obama}])$ = T *iff* (by definitions in (21))
- d. $[\lambda x. x \text{ smokes}](\text{Obama})$ = T *iff* (by definition in (20b))
- e. Obama smokes.

What just happened:

- In the preceding section, we first showed how a compositional extensional semantics can, for a given world/situation, compute the truth value of the sentence in that world/situation
- *Conversely*, if we take as *hypothesis* that the truth value of the sentence in the world is ‘TRUE’, our compositional extension semantics can ‘work backwards’, and compute *how the world must be constituted in order for the sentence to be true!*
- **Thus, we can use our extensional semantic function “[[]]” to compute the truth conditions of sentences!!!**

(26) **The Big Upshot**

An ‘extensional semantics’ – a formal system that maps complex structures to their extensions in the world – can provide us with a theory of how our brains recursively map sentences to their truth conditions...

Thus (for a significant portion of human language), such an ‘extensional semantics’ is sufficient for a theory of natural language meaning!

(27) **Our Over-Arching Project, Redefined Again**

We wish to develop the *right* theory of the function “[[.]]”, by examining the truth-conditions of particular sentences of the languages. Such a theory will consist of:

- (a) Primitive statements for the lexical items (idioms) of the language
- (b) Rules for deriving the value of “[[]]” for larger structures from the value of “[[]]” for their component parts.

So our task is to adjust (a) and (b) until our theory predicts *exactly* the correct truth conditions for every sentence of the language!

4. Some Follow-Up Technical Points

4.1 On the Semantics of Lexical Items

In Section 3.2, we developed a theory of the semantic value of the intransitive verb ‘smoked’, whereby we identified its extension with a particular type of function (from entities to T-values).

It’s instructive to reflect on *how* we came to this conclusion:

(28) Determining the ‘Meanings’ of a Lexical Item L

- (a) Consider the truth-conditions of sentences in which L appears.
- (b) Consider the (already established) extensions of the other lexical items in these sentences.
- (c) Based on (a) and (b), develop an entry for L which would – in combination with the entries for the other words in the sentences (b) – correctly derive the truth conditions of the sentences it appears in (a).

Thus, to determine the ‘meaning’ of a lexical item within this general program for semantics, we don’t (say) ‘introspect’ our ‘concept’ for the word...

(e.g. ‘smoking’ means lighting tobacco on fire and inhaling it, *etc. etc.*)

... rather, we look *only* to how the word contributes systematically to the truth conditions of a sentence.

- As we will see, this kind of an approach works especially well for the semantic analysis of *function words* like ‘and’, ‘the’, ‘every’, *etc.*
- For *content words* like ‘smoke’ and ‘child’, however, this approach admittedly ignores much of their intuitive ‘meaning’

(*cf.* the entry in (21) doesn’t say anything about what distinguishes ‘smoking’ from just ‘inhaling smoke’.)

- *This rather ‘sparse’ treatment of the semantics of ‘content words’ may or may not be a critical weakness of the overall ‘truth-conditional’ approach to semantics.*

(*cf.* Jackendoff 1990; Lakoff 1990)

4.2 More on the Lambda Notation

When we first introduced the ‘lambda notation’ for functions in (20b), our use of the notation was somewhat limited. It consisted of the following:

(29) Lambda Notation, Part 1

- a. *Syntax:* $\lambda x. \text{Predicate}(x)$
- b. *Semantics:* Function from entities to truth values, which when given an argument a yields T *iff* $\text{Predicate}(a)$
- c. *Example:*
 $\lambda x. x \text{ smokes.}$ ‘Function that takes an entity x and yields T *iff* x smokes.’

However, there are two general ways in which our extensional semantics will make a broader use of this lambda-notation.

(30) Functions that Don’t Take Entities as Arguments

- a. The functions characterized in (29) all have *entities* as their domain.
- b. However, as we’ll review, the extensions of natural language expressions are often functions whose domains are things *other than* entities (such as truth-values)
- c. We can indicate what kind of *thing* is in the domain of the function via a subscript on the variable that follows the lambda.
- d. Example: The Semantics of “It isn’t true that”

$$[[\text{it isn't true that}]] = \lambda p_t . p \neq T$$

To Prove: “It isn’t true that Obama smokes” is T *iff* Obama doesn’t smoke.

- i. $[[\text{It isn't true that Obama smokes}]] = T$ *iff* (by Rule (22))
- ii. $[[\text{it isn't true that}]]([[\text{Obama smokes}]]) = T$ *iff* (by (30d))
- iii. $[\lambda p_t . p \neq T]([[\text{Obama smokes}]]) = T$ *iff* (by definition of λ -notation)
- iv. $[[\text{Obama smokes}]] \neq T$ *iff* (by definitions in (21))
- v. $[\lambda x. x \text{ smokes}](\text{Obama}) \neq T$ *iff* (by definition in (20b))
- vi. Obama doesn’t smoke.

(31) **Functions that Don't Yield Truth Values**

- a. The functions characterized in (29) all yield *truth values* as their range.
- b. However, as we'll review, the extensions of natural language expressions are often functions whose range contains *other functions*.
- c. We can represent such functions in our lambda notation as follows:

Syntax: $\lambda x [\lambda y \dots \text{Predicate}(x) \dots]$

Semantics: The function which takes an argument a and yields the function
 $[\lambda y \dots \text{Predicate}(a) \dots]$

We can use 'function-valued-functions' like those defined in (31) to provide a semantics for (e.g.) *transitive verbs* (as you will recall).

(32) **The Semantics of 'Love'**

$[[\text{loves}]]$ = $\lambda y [\lambda x . x \text{ loves } y]$
The function which takes an argument b , and yields the function
Which takes an argument a and returns 'T' iff a loves b .

To Prove: "Dave loves Ann" is T iff Dave loves Ann

- a. $[[\text{Dave loves Ann}]]$ = T iff (by Rule (22), 'Function Application')
- b. $[[\text{loves Ann}]]([[\text{Dave}]])$ = T iff (by 'Function Application')
- c. $[[[\text{loves}]]] ([[\text{Ann}]]) ([[\text{Dave}]])$ = T iff (by (32) and other lexical entries)
- d. $[\lambda y [\lambda x . x \text{ loves } y]] (\text{Ann}) (\text{Dave})$ = T iff (by the definition of our λ -notation)
- e. $\lambda y [\lambda x . x \text{ loves Ann}] (\text{Dave})$ = T iff (by the definition of our λ -notation)
- f. Dave loves Ann

A Technical Side-Note:

There is a *prima facie* ambiguity in our lambda-notation:

- If there is no other lambda to the right of a lambda-operator, you read the formula ' $\lambda x. \text{Predicate}(x)$ ' as: 'The function that takes x and gives T iff $\text{Predicate}(x)$ '
- If there is another lambda to the right of a lambda operator, you read the formula ' $\lambda x. \lambda y. \dots \text{Predicate}(x) \dots$ ' as: 'The function that takes x and gives the function ' $\lambda y. \dots \text{Predicate}(x) \dots$ '

If one explores the formal foundations of 'lambda calculus', you'll find that this apparent ambiguity is only apparent, but it's something to bear in mind when reading formulae...

4.3 On Identifying ‘Truth Values’ as the Extensions of Sentences

Our semantic system has two properties that may at first strike the causal observer as ‘odd’ (and so we spent a good amount of time motivating them):

(33) Two Central (But Initially Confusing) Assumptions of Our Semantic Theory

- a. Sentences of natural language have *extensions* (‘denotations’, ‘reference’), and the *extension* of such sentences is a *truth value*.
- b. Our ‘semantic assignment function’ “[[.]]” maps the structures of natural language to their *extensions* (not, say, to the conceptual content (intensions) that determine those extensions).

We’ve already spent a good amount of time motivating the (initially odd) assumptions in (33) on *conceptual* grounds, by showing how they nevertheless contribute to our goal of having a semantic theory that derives truth-conditions.

However, it can also be said that there are empirical arguments for these two assumptions.

(34) Empirical Argument for the Assumptions in (33)

There are a variety of natural language expressions (e.g. the ‘logical connectives’) which:

- (i) *Syntactically, takes sentences as complements / specifiers / sisters*
- (ii) *Semantically, seem to take truth values as arguments.*

If the ‘meaning’ of a lexical item must take as argument the ‘meanings’ of its complements/specifiers, such ‘logical connectives’ *independently* show that:

- At some level, the ‘meaning’ of a sentence is its truth value.

This (still initially surprising) result makes the most sense within our general theory of intensions, extensions and “[[.]]”...

(35) A Logical Connective (‘Truth Functional Operator’) in Natural Language

$$[[\text{and}]] = \lambda p_t [\lambda q_t [p = T \text{ and } q = T]]$$

The extension of ‘and’ is a function which takes as argument a truth-value p and a truth-value q , and yields the value T iff p is T and q is T

As shown below, this semantics for the ‘logical connective’ “and” is sufficient to derive the truth-conditions of the sentences which contain it.

(36) **Deriving the Truth-Conditions of Conjoined Sentences**

To prove: “Obama smokes and Dave dances” is T iff Obama smokes and Dave dances.

- a. $[[\text{Obama smokes} [\text{and} [\text{Dave dances}]]]]$ is T iff (by F(unction) A(pplication))
- b. $[[\text{and} [\text{Dave dances}]]([[\text{Obama smokes}]])$ is T iff (by FA)
- c. $[[[\text{and}]]([[\text{Dave dances}]]) ([[\text{Obama smokes}]])$ is T iff (by (35))
- d. $[\lambda p_t [\lambda q_t [p = T \text{ and } q = T]] ([[\text{Dave dances}]]) ([[\text{Obama smokes}]])$ is T iff
(by definition of λ -notation)
- e. $[\lambda q_t [[[\text{Dave dances}]] = T \text{ and } q = T] ([[\text{Obama smokes}]])$ is T iff
(by definition of λ -notation)
- f. $[[\text{Dave dances}]] = T \text{ and } [[\text{Obama smokes}]] = T$ iff (by FA and lexical entries)
- g. Dave dances and Obama smokes.

General Point:

- For a large number of structural types, things really do work out well if we assume that:
 - (a) the extensions of sentences are *truth values*.
 - (b) the semantic assignment function “[]” yields the extension of a given structure

5. Type Theory and Type-Driven Interpretation

The last thing to cover in this initial ‘fundamentals’ portion of the review is the theory of ‘semantic types’ and ‘type-driven interpretation’.

Thus far, our simple semantic system has used the following kinds (or ‘types’) of things as extensions:

(37) **The Kinds (‘Types’) of Extensions our System Uses**

- | | | |
|----|---|--|
| a. | Entities | $[[\text{Obama}]]$ = Obama |
| b. | Truth Values | $[[\text{Obama smokes}]]$ = True |
| c. | Functions from entities to truth values: | $[[\text{smokes}]]$ = $[\lambda x. x \text{ smokes}]$ |
| d. | Functions from entities to functions: | $[[\text{loves}]]$ = $[\lambda y. [\lambda x x \text{ loves } y]]$ |
| e. | Functions from truth values to truth values | $[[\text{and}]]$ = $[\lambda p [\lambda q [p = T \text{ and } q = T]]]$ |

(43) **Why is This Vocabulary of ‘Semantic (Logical) Types’ Useful?**

- a. It simplifies the statement of generalizations about ‘lexical entries’
We can now simply say, for example, things like:
- (i) intransitive verbs (and VPs) are of type $\langle e \ t \rangle$
 - (ii) sentences are of type t
 - (iii) proper names are of type e
 - (iv) transitive verbs are of type $\langle e \ \langle e \ t \rangle \rangle$
 - (v) quantificational determiners are of type $\langle \langle e \ t \rangle \ \langle \langle e \ t \rangle \ t \rangle \rangle$
- b. It simplifies the statement of our semantic ‘rules’
Compare the original formulation of FA in (22) to the following:

- (i) *Function Application (New Formulation)*

If X is a structure consisting of two daughters – Y and Z – and if Y is of type $\langle \sigma \ \tau \rangle$ and Z is of type σ , then $[[X]] = [[Y]]([[Z]])$

To see the ‘breath-saving’ work of our ‘type theory’ in action, consider the following (new) semantic rule:

(44) **The Rule of ‘Predicate Modification’ (Heim & Kratzer 1998)**

If a phrase X is a structure consisting of two daughters – Y and Z – and if both Y and Z are of type $\langle \sigma \ \tau \rangle$, then $[[X]] = [\lambda x_{\sigma} . [[Y]](x) \ \& \ [[Z]](x)]$

What does this rule in (44) accomplish for us? As illustrated below, it can (perhaps) provide a compositional treatment of relative clause modification!

(45) **Background Assumptions Regarding Relative Clauses**

- a. Nouns are of Type $\langle e \ t \rangle$

$[[\text{boy}]]$ = $\lambda x . x \text{ is a boy}$

- b. Relative Clauses are of Type $\langle e \ t \rangle$

Relative clauses with subject gaps are semantically equivalent to the (type $\langle e \ t \rangle$) VPs that they contain.

$[[\text{that smokes}]]$ = $[[\text{smokes}]]$ = $\lambda x . x \text{ smokes.}$

$[[\text{that loves Dave}]]$ = $[[\text{loves Dave}]]$ = $\lambda x . x \text{ loves Dave.}$

(46) The Rule of ‘Predicate Modification’ (PM) in Action: Relative Clauses

- a. $[[\text{boy} [\text{that smokes}]]]$ = (by PM)
- b. $\lambda x. [[\text{boy}]](x) \ \& \ [[\text{that smokes}]](x)$ = (by (45))
- c. $\lambda x. [\lambda y . y \text{ is a boy}](x) \ \& \ [\lambda z. z \text{ smokes}](x)$ = (by definition of ‘ λx ’)
- c. $\lambda x. x \text{ is a boy} \ \& \ x \text{ smokes}$

When we develop our theory of determiners in the next set of review notes, we’ll see that this semantics for adnominal relatives captures some core aspects of their meaning!

Summary of What We’ve Reviewed Thus Far:

- Qua linguists, we want a formal model of the procedure our ‘mind/brains’ use to assign ‘interpretations’ to the structures of our languages.
- A system that seems to have a good shot of doing this is one that can derive the ‘truth conditions’ of sentences from the ‘meanings’ of their component parts.
- A system that seems to have a good shot of doing *that* is one that (i) computes the so-called ‘extensions’ of structures from the ‘extensions’ of their parts, and (ii) identifies the ‘extension’ of a sentence as its truth value.
- In such a system, the notion that ‘meanings combine together to create new meanings’ is captured via the modeling of some ‘meanings’ (extensions) as functions from objects of one type to objects of another.
- Thus, in such a system, the ‘combining’ of meanings is formally modeled as *function application*. (slogan, possibly false = ‘all semantic composition is function application’)
- The general (still maybe unclear) distinction between ‘extension’ and ‘intension’
- The ‘lambda notation’ for functions
- The theory of ‘semantic (logical) types’