

Homework Assignment 8

Any automatically graded answer may be manually graded by the instructor. Submissions are expected to only use functions taught in the course. If a submission uses a disallowed function, that exercise can get zero points. Excluding promises, *all functions that mutate values are disallowed* (mutable functions usually have a ! in their name).

Translating SimpleJS into LambdaJS

1. Implement the following translation function from SimpleJS into LambdaJS. LambdaJS variables are underlined, while SimpleJS are not. We use the abstract-syntax notation for the let-binding, sequencing, the object constructor, and the function declaration (λ). We use indentation to highlight the scope of let-binders and of sequencing. You are encouraged to peruse `hw8-util.rkt`, as it gives usage examples and has helpful documentation to complete this assignment.

$$\begin{aligned} J[x.y] &\stackrel{\text{def}}{=} (\text{get-field } (\text{deref } J[x]) \text{ "y"}) \\ J[x.y := e] &\stackrel{\text{def}}{=} \text{let } \underline{data} = J[e] \text{ in} \\ &\quad \text{let } \underline{o} = (\text{deref } J[x]) \text{ in} \\ &\quad (\text{set! } J[x] \text{ (update-field } \underline{o} \text{ "y" } \underline{data})) ; \\ &\quad \underline{data} \\ J[x.y(e \dots)] &\stackrel{\text{def}}{=} \text{let } \underline{m} = (\text{get-field } (\text{deref } J[x]) \text{ "y"}) \text{ in} \\ &\quad \text{let } \underline{f} = (\text{get-field } (\text{deref } \underline{m}) \text{ "$code"}) \text{ in} \\ &\quad (\underline{f} \ J[x] \ J[e \dots]) \\ J[\text{function}(x \dots) \{e\}] &\stackrel{\text{def}}{=} (\text{alloc } \{ \text{"$code"} : \lambda(\underline{this}, J[x] \dots). J[e], \text{"prototype"} : (\text{alloc } \{\}) \}) \\ J[\text{new } e_f(e \dots)] &\stackrel{\text{def}}{=} \text{let } \underline{ctor} = (\text{deref } J[e_f]) \text{ in} \\ &\quad \text{let } \underline{obj} = (\text{alloc } \{ \text{"$proto"} : (\text{get-field } \underline{ctor} \text{ "prototype"}) \}) \text{ in} \\ &\quad \text{let } \underline{f} = (\text{get-field } \underline{ctor} \text{ "$code"}) \text{ in} \\ &\quad (\underline{f} \ \underline{obj} \ J[e] \dots) ; \\ &\quad \underline{obj} \\ J[c] &\stackrel{\text{def}}{=} c \\ J[x] &\stackrel{\text{def}}{=} x \\ J[\text{let } x = e_1 \text{ in } e_2] &\stackrel{\text{def}}{=} \text{let } J[x] = J[e_1] \text{ in } J[e_2] \end{aligned}$$