In a mobile language, computations can move between locations in a network to better utilise resources. Mobile Haskell, or mHaskell, is a small extension of Concurrent Haskell that enables the construction of distributed mobile software by introducing higher order communication channels called Mobile Channels (MChannels). mHaskell only provides weak mobility, i.e. the ability to start new computations on remote locations. This paper shows how strong mobility, i.e. the ability to migrate running threads between locations, can be implemented in a language like mHaskell with weak mobility, higher-order channels and first-class continuations. Using Haskell's high level features, such as higher-order functions, type classes and support for monadic programming, strong mobility is achieved without any changes to the runtime system, or built-in support for continuations. Strong mobility is illustrated with examples and a mobile agent case study.