EIP 190: Ethereum Smart Contract Packaging Standard

Return to Ethereum ERCs

> **Note**: The following is an excerpt from the official Ethereum site. It is provided here as a convenience and is not authoritative. Refer to the original document as the authoritative reference.

Table 1: Data sheet for Ethereum Smart Contract Packaging Standard

Title	Ethereum Smart Contract Packaging Standard
Author	Piper Merriam, Tim Coulter, Denis Erfurt, RJ Catalano, Iuri Matias
Status	Final
Created	2017-01-10
Description	http://eips.ethereum.org/EIPS/eip-190
Specification	http://eips.ethereum.org/EIPS/eip-190#Specification
Category	ERC

Abstract

This ERC proposes a specification for Ethereum smart contract packages.

The specification was collaboratively developed by the following Ethereum development framework maintainers.

- *Tim Coulter (Truffle)*
- *Denis Erfurt (Dapple)*
- *Piper Merriam (Populus)*
- *RJ Catalano (Eris PM)*
- *Iuri Matias (Embark)*

Motivation

Packaging is a core piece of modern software development which is missing from the Ethereum ecosystem. The lack of packaging limits the ability for developers to reuse code which negatively affects productivity and security.

A key example of this is the ERC20 standard. There are a few well audited reusable token contracts available but most developers end up writing their own because of the difficulty in finding and reusing existing code.

A packaging standard should have the following positive effects on the ecosystem:

- *Greater overall productivity caused by the ability to reuse existing code.*
- *Increased security caused by the ability to reuse existing well audited implementations of common patterns (ERC20, crowdfunding, etc).*

Smart contract packaging should also have a direct positive effect on the end user. Wallet software will be able to consume a released package and generate an interface for interacting with any deployed contracts included within that package. With the advent of ENS all of the pieces will be in place for a wallet to take a human readable name and present the user with an interface for interacting with the underlying application.