Difference between Loaders and Plugins in Webpack

Loaders	Plugins

• Loaders in webpack are transformations that are applied to the source code of a module. They allow you to pre-process files as you require() or import them.
 	
• Loaders are typically used to transform files from one format to another, such as converting TypeScript to JavaScript, or Sass/SCSS to CSS.
 	
• Loaders are defined in the module.rules array in the webpack configuration file
 	
• Installing Style loader and CSS loader:
npm i -D style-loader css-loader
• Example:
module: {
rules: [
{
test: /\.css$/,
use: ["style-loader", "css-loader",”less-loader”]
}
]
},
 	
In the above example,
• the less-loader will convert the file into CSS.
• then the css-loader will convert into a javascript file in an array format.
• which can be consumed by style-loader and -style-loader converts the file into a JavaScript module.
	• Plugins in webpack are more powerful than loaders. They can be used to perform a wider range of tasks like bundle optimization, asset management, and injection of environment variables.
	
	• Plugins have access to the entire webpack lifecycle and can interact with the compiler.
	
	• Plugins are typically instantiated using the new keyword and are included in the plugins array in the webpack configuration file.
	
	• 80% of the webpack is made up of its own plugin system. Webpack itself is an event-driven architecture.
	
	• Installing plugins
	• npm i -D mini-css-extract-plugin
	• Example:
	const HtmlWebpackPlugin = require("html-webpack-plugin");
	
	module.exports = {
	plugins: [
	new HtmlWebpackPlugin({
	template: "./src/index.html",
	}),
	],
	};
	In the above example
	In this example, HtmlWebpackPlugin is a plugin used to generate an HTML file and inject the bundled JavaScript automatically.