<span class="hljs-keyword">export</span> <span class="hljs-function"><span class="hljs-keyword">function</span> <span class="hljs-title">getModuleInstanceState</span>(<span class="hljs-params">node: Node</span>): <span class="hljs-title">ModuleInstanceState</span> </span>{

<span class="hljs-comment">// A module is uninstantiated if it contains only</span>

<span class="hljs-comment">// 1. interface declarations, type alias declarations</span>

<span class="hljs-keyword">if</span> (node.kind === SyntaxKind.InterfaceDeclaration || node.kind === SyntaxKind.TypeAliasDeclaration) {

<span class="hljs-keyword">return</span> ModuleInstanceState.NonInstantiated;

}

<span class="hljs-comment">// 2. const enum declarations</span>

<span class="hljs-keyword">else</span> <span class="hljs-keyword">if</span> (isConstEnumDeclaration(node)) {

<span class="hljs-keyword">return</span> ModuleInstanceState.ConstEnumOnly;

}

<span class="hljs-comment">// 3. non-exported import declarations</span>

<span class="hljs-keyword">else</span> <span class="hljs-keyword">if</span> ((node.kind === SyntaxKind.ImportDeclaration || node.kind === SyntaxKind.ImportEqualsDeclaration) &amp;&amp; !(node.flags &amp; NodeFlags.Export)) {

<span class="hljs-keyword">return</span> ModuleInstanceState.NonInstantiated;

}

<span class="hljs-comment">// 4. other uninstantiated module declarations.</span>

<span class="hljs-keyword">else</span> <span class="hljs-keyword">if</span> (node.kind === SyntaxKind.ModuleBlock) {

<span class="hljs-keyword">let</span> state = ModuleInstanceState.NonInstantiated;

forEachChild(node, <span class="hljs-function"><span class="hljs-params">n</span> =&gt;</span> {

<span class="hljs-keyword">switch</span> (getModuleInstanceState(n)) {

<span class="hljs-keyword">case</span> ModuleInstanceState.NonInstantiated:

<span class="hljs-comment">// child is non-instantiated - continue searching</span>

<span class="hljs-keyword">return</span> <span class="hljs-literal">false</span>;

<span class="hljs-keyword">case</span> ModuleInstanceState.ConstEnumOnly:

<span class="hljs-comment">// child is const enum only - record state and continue searching</span>

state = ModuleInstanceState.ConstEnumOnly;

<span class="hljs-keyword">return</span> <span class="hljs-literal">false</span>;

<span class="hljs-keyword">case</span> ModuleInstanceState.Instantiated:

<span class="hljs-comment">// child is instantiated - record state and stop</span>

state = ModuleInstanceState.Instantiated;

<span class="hljs-keyword">return</span> <span class="hljs-literal">true</span>;

}

});

<span class="hljs-keyword">return</span> state;

}

<span class="hljs-keyword">else</span> <span class="hljs-keyword">if</span> (node.kind === SyntaxKind.ModuleDeclaration) {

<span class="hljs-keyword">return</span> getModuleInstanceState((&lt;ModuleDeclaration&gt;node).body);

}

<span class="hljs-keyword">else</span> {

<span class="hljs-keyword">return</span> ModuleInstanceState.Instantiated;

}

}