

gleam-lang / gleam Public

<> Code Issues 162 Pull requests 52 Discussions Actions Projects

Commit 1aa5d8e

 lpil committed 4 days ago · ✓ 14 / 14

Error immediately for invalid config

main · v1.16.0-rc1
1 parent [b30d200](#) commit 1aa5d8e

4 files changed +128 -10 lines changed

↑ Top ⚙️

Filter files...

CHANGELOG.md

compiler-core/src

config.rs

snapshots

gleam_core__config__invalid_dependency_name.snap

gleam_core__config__invalid_dev_dependency_name.snap

4 files changed +128 -10 lines changed

Search within code ⚙️

CHANGELOG.md

<> 📄 ⋮

```

@@ -173,6 +173,10 @@
173 173 - New Gleam packages are generated requiring >= 0.70.0 of `gleam_stdlib`.
174 174   ([Louis Pilfold](https://github.com/lpil))
175 175
176 + - `gleam.toml` files with invalid dependency names now raise an error
177 +   immediately when the file is parsed.
178 +   ([Louis Pilfold](https://github.com/lpil))
179 +

```

```

176 180 - Documentation for `--target` option has been improved to include more
177 181 details.
178 182 ([Andrey Kozhev](https://github.com/ankddev))

```



▼ compiler-core/src/config.rs



```
@@ -163,9 +163,18 @@ pub struct PackageConfig {
```

```

163 163     pub description: EcoString,
164 164     #[serde(default, alias = "docs")]
165 165     pub documentation: Docs,
166 166 -     #[serde(default, serialize_with = "ordered_map")]
166 166 +     #[serde(
167 167 +         default,
168 168 +         serialize_with = "ordered_map",
169 169 +         deserialize_with = "dependencies_map::deserialize"
170 170 +     )]
171 171     pub dependencies: Dependencies,
172 172 +     #[serde(
173 173 +         default,
174 174 +         alias = "dev-dependencies",
175 175 +         serialize_with = "ordered_map",
176 176 +         deserialize_with = "dependencies_map::deserialize"
177 177 +     )]
178 178     pub dev_dependencies: Dependencies,
179 179     #[serde(default)]
180 180     pub repository: Option<Repository>,
181 181
182 182 @@ -362,6 +371,34 @@ aide_generator = { git =
183 183     "git@github.com:crowdhaier/aide.git", ref = "f559c5bc"
184 184
185 185     insta::assert_snapshot!(insta::internals::AutoName,
186 186     error.pretty_string());
187 187 }
188 188
189 189
190 190 + #[test]
191 191 + fn invalid_dependency_name() {
192 192 +     let toml = r#"
193 193 +     name = "wibble"
194 194 +     version = "1.0.0"
195 195 +
196 196 +

```

```

380 + [dependencies]
381 + Hello = "> 1.0.0"
382 + "#;
383 +     let error = deserialise_config("gleam.toml", toml.into())
384 +         .expect_err("should fail to deserialise because invalid name");
385 +     insta::assert_snapshot!(insta::internals::AutoName,
    error.pretty_string());
386 + }
387 +
388 + #[test]
389 + fn invalid_dev_dependency_name() {
390 +     let toml = r#"
391 +     name = "wibble"
392 +     version = "1.0.0"
393 +
394 +     [dev_dependencies]
395 +     "yes." = "> 1.0.0"
396 +     "#;
397 +     let error = deserialise_config("gleam.toml", toml.into())
398 +         .expect_err("should fail to deserialise because invalid name");
399 +     insta::assert_snapshot!(insta::internals::AutoName,
    error.pretty_string());
400 + }
401 +

```

```

365 402 #[test]
366 403 fn locked_no_manifest() {
367 404     let mut config = PackageConfig::default();

```



```
@@ -1067,11 +1104,8 @@ mod uri_serde_default_https {
```

```

1067 1104
1068 1105     mod package_name {
1069 1106         use ecow::EcoString;
1070 -     use regex::Regex;
1071 1107         use serde::Deserializer;
1072 -     use std::{fmt, sync::OnceLock};
1073 -
1074 -     static PACKAGE_NAME_PATTERN: OnceLock<Regex> = OnceLock::new();
1108 +     use std::fmt;
1075 1109

```

```

1076 1110     pub fn deserialize<'de, D>(serializer: D) -> Result<EcoString,
        D::Error>
1077 1111     where
@@ -1093,10 +1127,7 @@ mod package_name {
1093 1127         where
1094 1128             E: serde::de::Error,
1095 1129         {
1096 -         if PACKAGE_NAME_PATTERN
1097 -             .get_or_init(|| Regex::new("[a-z][a-z0-
9_]*$").expect("Package name regex"))
1098 -             .is_match(value)
1099 -             {
1130 +         if super::is_valid_package_name(value) {
1100 1131             Ok(value.into())
1101 1132         } else {
1102 1133             let error =
@@ -1107,6 +1138,53 @@ mod package_name {
1107 1138         }
1108 1139     }
1109 1140
1141 + mod dependencies_map {
1142 +     use ecow::EcoString;
1143 +     use serde::{Deserialize, Deserializer, de};
1144 +     use std::collections::HashMap;
1145 +
1146 +     pub fn deserialize<'de, D, V>(serializer: D) ->
Result<HashMap<EcoString, V>, D::Error>
1147 +     where
1148 +         D: Deserializer<'de>,
1149 +         V: Deserialize<'de>,
1150 +     {
1151 +         struct DependenciesVisitor<V>(std::marker::PhantomData<V>);
1152 +
1153 +         impl<'de, V: Deserialize<'de>> de::Visitor<'de> for
DependenciesVisitor<V> {
1154 +             type Value = HashMap<EcoString, V>;
1155 +
1156 +             fn expecting(&self, f: &mut std::fmt::Formatter<'_>) ->
std::fmt::Result {
1157 +                 write!(f, "a map with valid package name keys")

```

```
1158 +     }
1159 +
1160 +     fn visit_map<A: de::MapAccess<'de>>(self, mut map: A) ->
Result<Self::Value, A::Error> {
1161 +         let mut dependencies = HashMap::new();
1162 +         while let Some(key) = map.next_key::<EcoString>()? {
1163 +             if !super::is_valid_package_name(&key) {
1164 +                 return Err(de::Error::invalid_value(
1165 +                     de::Unexpected::Str(&key),
1166 +                     &"a package name containing only lowercase
letter, numbers, and underscores",
1167 +                     ));
1168 +             }
1169 +             let _: Option<_> = dependencies.insert(key,
map.next_value()?);
1170 +         }
1171 +         Ok(dependencies)
1172 +     }
1173 + }
1174 +
1175 +     deserializer.deserialize_map(DependenciesVisitor(std::marker::PhantomData))
1176 + }
1177 + }
1178 +
1179 + fn is_valid_package_name(name: &str) -> bool {
1180 +     use regex::Regex;
1181 +     use std::sync::OnceLock;
1182 +     static PACKAGE_NAME_PATTERN: OnceLock<Regex> = OnceLock::new();
1183 +     PACKAGE_NAME_PATTERN
1184 +         .get_or_init(|| Regex::new("[a-z][a-z0-9_]*").expect("Package name
regex"))
1185 +         .is_match(name)
1186 + }
1187 +
1110 1188 #[test]
1111 1189 fn name_with_dash() {
1112 1190     let input = r#"

```



▼ ...core_config_invalid_dependency_name.snap

```

... @@ -0,0 +1,18 @@
1 + ---
2 + source: compiler-core/src/config.rs
3 + assertion_line: 385
4 + expression: error.pretty_string()
5 + ---
6 + error: File IO failure
7 +
8 + An error occurred while trying to parse this file:
9 +
10 +     gleam.toml
11 +
12 + The error message from the file IO library was:
13 +
14 +     TOML parse error at line 5, column 1
15 +   |
16 + 5 | [dependencies]
17 +   | ^^^^^^^^^^^^^^^
18 + invalid value: string "Hello", expected a package name containing only lowercase
    letter, numbers, and underscores

```

▼ ...e_config_invalid_dev_dependency_name.snap

```

... @@ -0,0 +1,18 @@
1 + ---
2 + source: compiler-core/src/config.rs
3 + assertion_line: 399
4 + expression: error.pretty_string()
5 + ---
6 + error: File IO failure
7 +
8 + An error occurred while trying to parse this file:
9 +
10 +     gleam.toml
11 +
12 + The error message from the file IO library was:
13 +
14 +     TOML parse error at line 5, column 1

```

```
15 + |
16 + 5 | [dev_dependencies]
17 + | ^^^^^^^^^^^^^^^^^^^^^^^^^
18 + invalid value: string "yes.", expected a package name containing only lowercase
    letter, numbers, and underscores
```

Comments 0



Please [sign in](#) to comment.