

Non-extensible Applies to Private for Stage 3

Mark S. Miller



Shu-yu Guo



Chip Morningstar



Erik Marks



110th Plenary
September 2025

TC
39

Recap:

1.1 PrivateFieldAdd (*O*, *P*, *value*)

The abstract operation PrivateFieldAdd takes arguments *O* (an Object), *P* (a Private Name), and *value* (an ECMAScript language value) and returns either a normal completion containing UNUSED or a throw completion. It performs the following steps when called:

1. If *O*.[[Extensible]] is **false**, throw a **TypeError** exception.
2. If the *host* is a web browser, then
 - a. Perform ? HostEnsureCanAddPrivateElement(*O*).
3. Let *entry* be PrivateElementFind(*O*, *P*).
4. If *entry* is not EMPTY, throw a **TypeError** exception.
5. Append PrivateElement { [[Key]]: *P*, [[Kind]]: FIELD, [[Value]]: *value* } to *O*.[[PrivateElements]].
6. Return UNUSED.

1.2 PrivateMethodOrAccessorAdd (*O*, *method*)

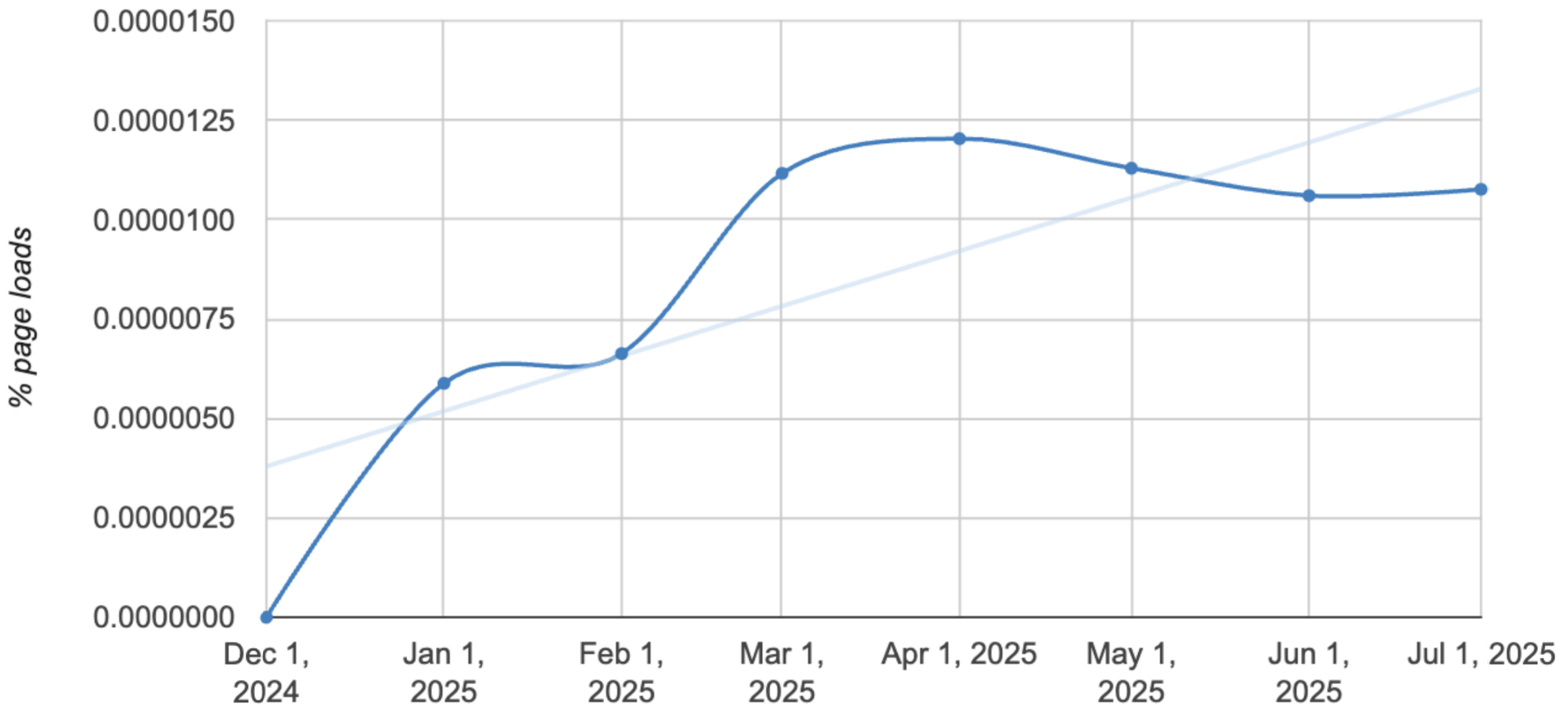
The abstract operation PrivateMethodOrAccessorAdd takes arguments *O* (an Object) and *method* (a PrivateElement) and returns either a normal completion containing UNUSED or a throw completion. It performs the following steps when called:

1. **Assert**: *method*.[[Kind]] is either METHOD or ACCESSOR.
2. If *O*.[[Extensible]] is **false**, throw a **TypeError** exception.
3. If the *host* is a web browser, then
 - a. Perform ? HostEnsureCanAddPrivateElement(*O*).
4. Let *entry* be PrivateElementFind(*O*, *method*.[[Key]]).
5. If *entry* is not EMPTY, throw a **TypeError** exception.
6. Append *method* to *O*.[[PrivateElements]].
7. Return UNUSED.

Recap:

Percentage of page loads over time

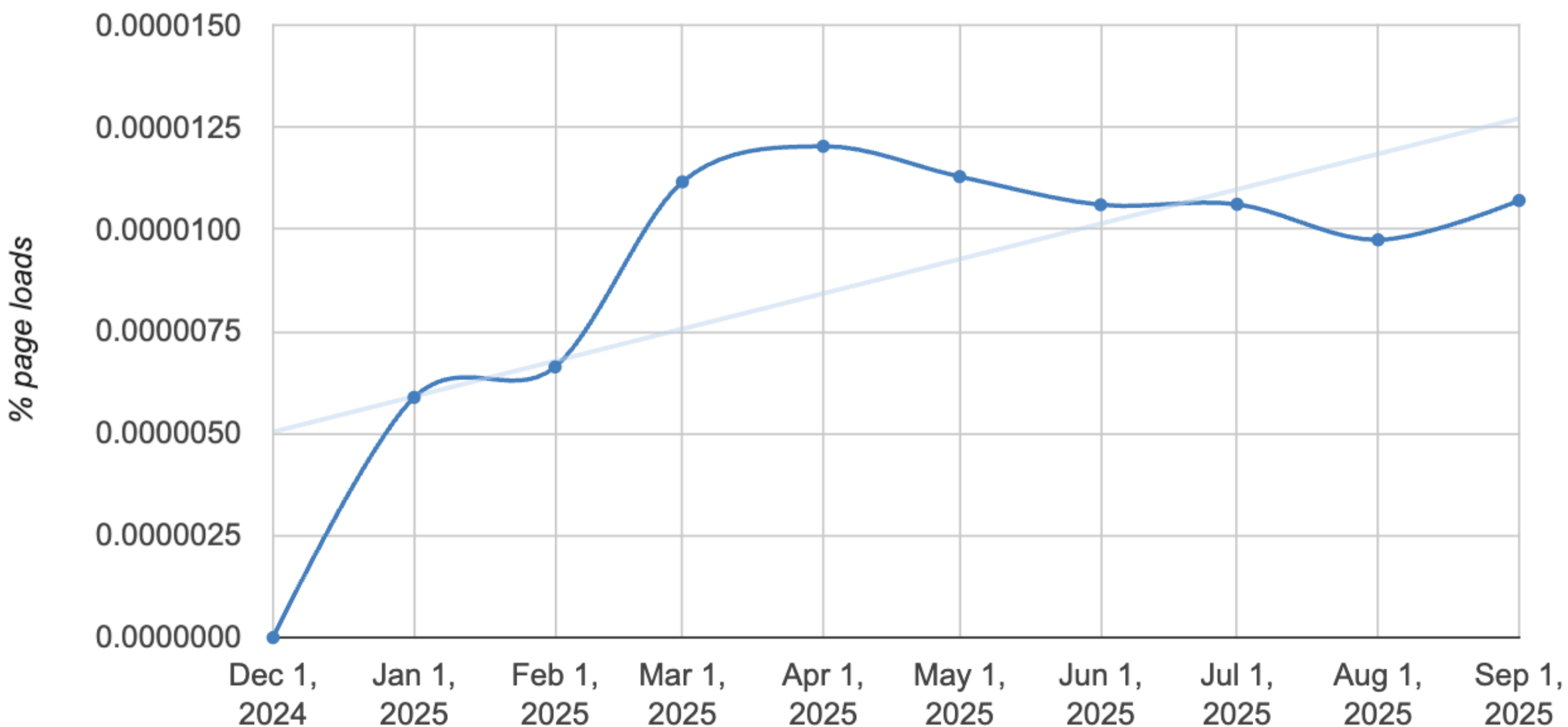
The chart below shows the percentage of page loads (in Chrome) that use this feature at least once. Data is across all channels and platforms. Newly added use counters that are not on Chrome stable yet only have data from the Chrome channels they're on.



Recap++:

Percentage of page loads over time

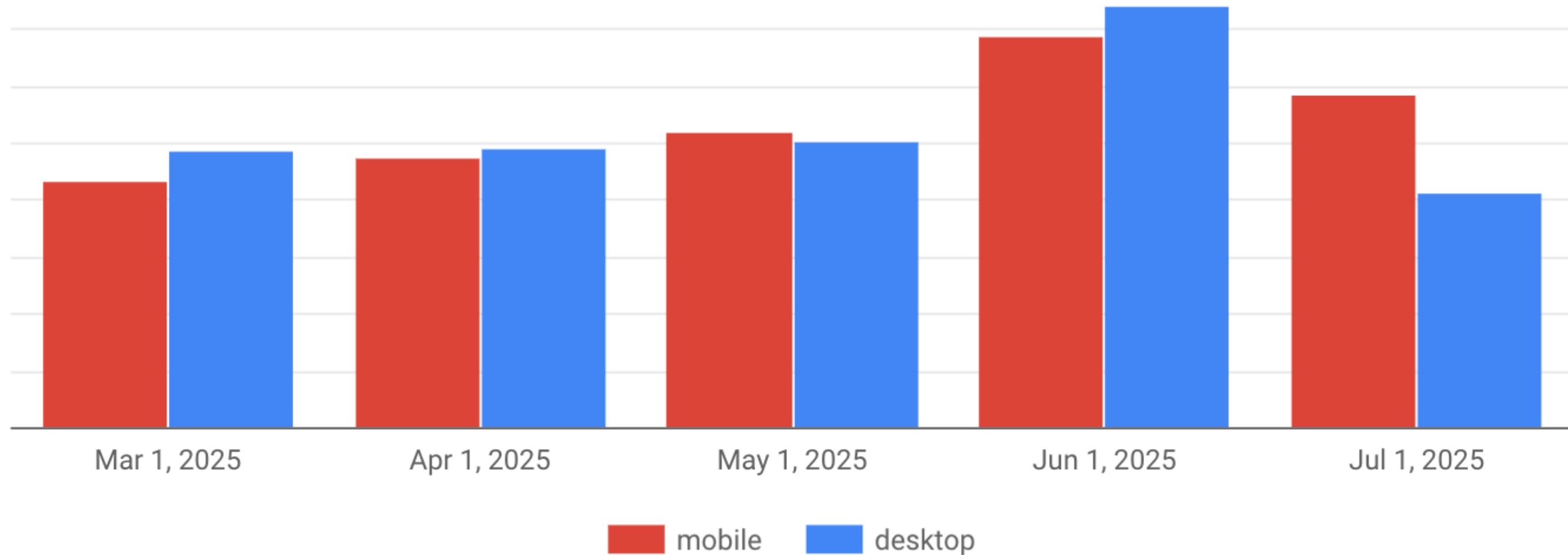
The chart below shows the percentage of page loads (in Chrome) that use this feature at least once. Data is across all channels and platforms. Newly added use counters that are not on Chrome stable yet only have data from the Chrome channels they're on.



Recap:

Adoption of the feature on top sites

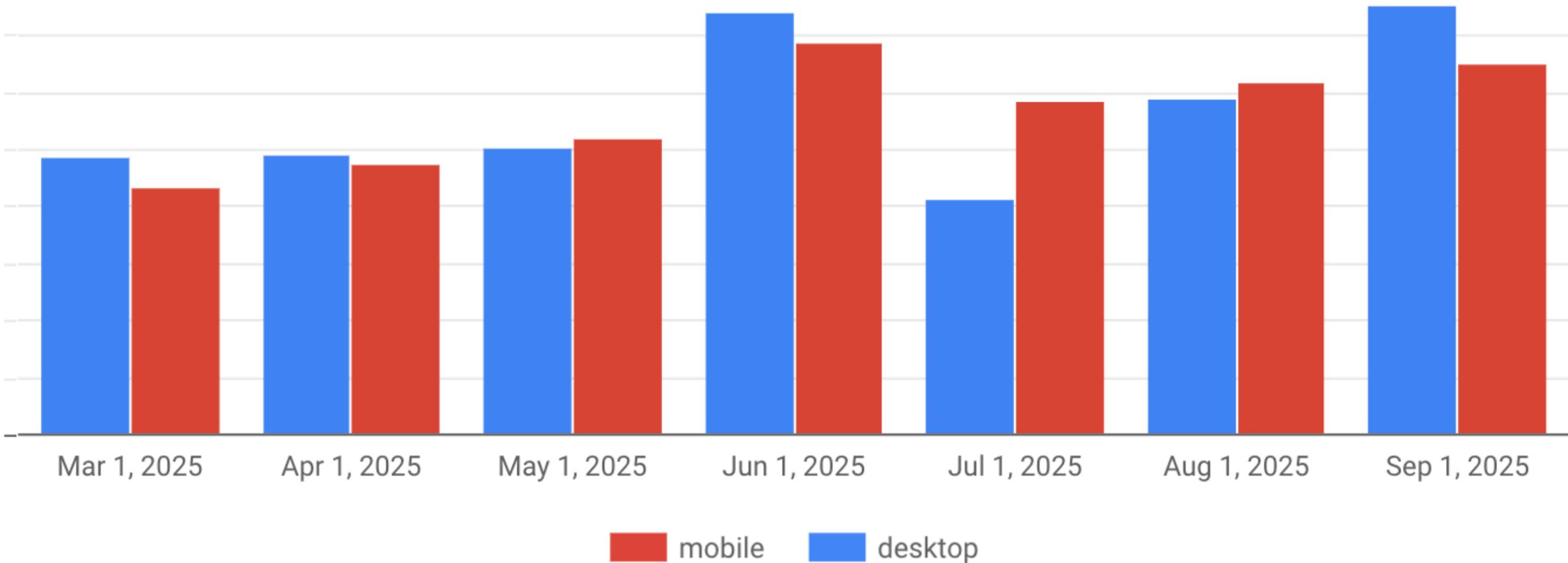
The chart below shows the adoption of the feature by the top URLs on the internet. Data from [HTTP Archive](#).



Recap++:

Adoption of the feature on top sites

The chart below shows the adoption of the feature by the top URLs on the internet. Data from [HTTP Archive](#).



Recap: from Babel downleveling



Jamesernator on Apr 18

...

The minified version looks like the following.

Given the field is only used for the initializer, there's a good chance it comes from Babel's downleveling of static blocks as that exhibits [basically this exact downleveling](#).



nicolo-ribaudo

...

```
export class C {  
  static #F00 = "#F00";  
  static {  
    C.#F00 = #BAR in C;  
  }  
  static #BAR = C.#F00;  
  static {  
    Object.freeze(C);  
  }  
}
```

```
var lastBlocks;  
export class C {  
  static #F00 = "#F00";  
  static #BAR = (  
    (() => {  
      C.#F00 = #BAR in C;  
    })(),  
    lastBlocks = (() => {  
      Object.freeze(C);  
    }),  
    C.#F00  
  );  
}  
lastBlocks();
```

Recap: Only test262 approvals needed

for stage 3

from previous plenary stage 2.7 update

https://www.youtube.com/watch?v=kFBvqdQOjQ8&list=PLzDw4TTug5O3vIAd4IR1Gp5t_46co_dv9



erights commented 3 days ago

Member

Author



Looks correct so far.

For completeness, would we need to test stamping a private accessor and private method?

For inheritance, done.

For a base class, AFAICT, impossible. There are no observable states where the class, instance, or prototype are observable before methods or accessors are added.

(I would guess that the case of stamping a private static method, or private static field cannot occur, since those go on the constructor, which cannot be frozen prior to the class declaration.)

For a private static method, AFAICT, it is indeed impossible.

For a private static field, it can be done in the initializer expression. Done.

Finally, this needs a feature flag for the proposal to be added in `features.txt` and in the frontmatter of the two modified tests.

Done.

(with much help from [@gibson042](#))

Thanks also to @o- Olivier Flückiger

PTAL



erights requested a review from **ptomato** 3 days ago



ptomato approved these changes on behalf of **tc39/test262-maintainers** 3 days ago

[View reviewed changes](#)

ptomato left a comment

Member



As far as I can tell complete now, thanks!



Questions?

1.1 PrivateFieldAdd (*O*, *P*, *value*)

The abstract operation PrivateFieldAdd takes arguments *O* (an Object), *P* (a Private Name), and *value* (an ECMAScript language value) and returns either a normal completion containing UNUSED or a throw completion. It performs the following steps when called:

1. If *O*.[[Extensible]] is **false**, throw a **TypeError** exception.
2. If the *host* is a web browser, then
 - a. Perform ? HostEnsureCanAddPrivateElement(*O*).
3. Let *entry* be PrivateElementFind(*O*, *P*).
4. If *entry* is not EMPTY, throw a **TypeError** exception.
5. Append PrivateElement { [[Key]]: *P*, [[Kind]]: FIELD, [[Value]]: *value* } to *O*.[[PrivateElements]].
6. Return UNUSED.

1.2 PrivateMethodOrAccessorAdd (*O*, *method*)

The abstract operation PrivateMethodOrAccessorAdd takes arguments *O* (an Object) and *method* (a PrivateElement) and returns either a normal completion containing UNUSED or a throw completion. It performs the following steps when called:

1. Assert: *method*.[[Kind]] is either METHOD or ACCESSOR.
2. If *O*.[[Extensible]] is **false**, throw a **TypeError** exception.
3. If the *host* is a web browser, then
 - a. Perform ? HostEnsureCanAddPrivateElement(*O*).
4. Let *entry* be PrivateElementFind(*O*, *method*.[[Key]]).
5. If *entry* is not EMPTY, throw a **TypeError** exception.
6. Append *method* to *O*.[[PrivateElements]].
7. Return UNUSED.