

Le monstre RUM





Les données RUM, qu'est-ce que c'est?

RUM: une définition

Les données Real User Monitoring (RUM) de performance sont les mesures collectées directement sur les visiteurs d'un site web, de façon passive.

Ces mesures proviennent typiquement d'APIs JavaScript disponibles côté client, comme [Navigation Timing](#) ou [Resource Timing](#), qui sont collectées au niveau du navigateur et envoyées côté serveur, la plupart du temps via l'API [Beacon](#).

Mesures RUM et synthétiques

Certaines mesures sont uniquement pertinentes si collectées en RUM. Certaines ne peuvent être mesurées qu'en tests synthétiques.

Parfois il est intéressant de les collecter dans les deux cas. Voici quelques exemples.

Mesure	Pertinent en RUM	Pertinent en synthétique
SpeedIndex	Impossible à mesurer	Oui
Networking API	Oui	Non
Navigation Timing API	Oui	Oui
Resource Timing API	Oui	Oui
Paint Timing API	Oui	Oui

Le RUM peut aussi être sur-mesure

Exemple: à l'aide d'un Worker nous mesurons actuellement la puissance CPU disponible de l'appareil. Score qui peut être très différent entre deux appareils du même modèle (niveau de batterie, parallélisme du système d'exploitation, etc.).

Grâce à [Element Timing](#), il est désormais possible de mesurer l'affichage d'éléments particuliers (logo, image "héros", appel à action, etc.).



Un domaine en
constante évolution

Layout Stability

Mesure quand du contenu se fait pousser visuellement après son apparition initiale.

Nous avons participé à l'[Origin Trial](#) de Chrome d'avril à Septembre.

Nous avons découvert 2 bugs dans cette version de test qui ont été résolus depuis. A présent cette API nous permet de découvrir de vrais problèmes qui étaient jusque là difficiles à capturer.



Illustration: Steve Kobes, Google

Event Timing

Mesure quand un événement utilisateur (eg. clic) est traité trop lentement par l'application. Le traitement synchrone de ces événements bloque le navigateur, ce qui donne une sensation d'interface gelée.

Grâce à notre participation à l'Origin Trial, nous avons découvert des problèmes importants et récurrents sur Wikipedia, qui sont en cours de traitement.

Une cause fréquente une utilisation excessive de fonctions qui font recalculer les styles pendant le traitement d'un événement.

Element Timing

Enfin un moyen de mesurer quand l'élément qu'on souhaite observer apparaît réellement sur l'écran!

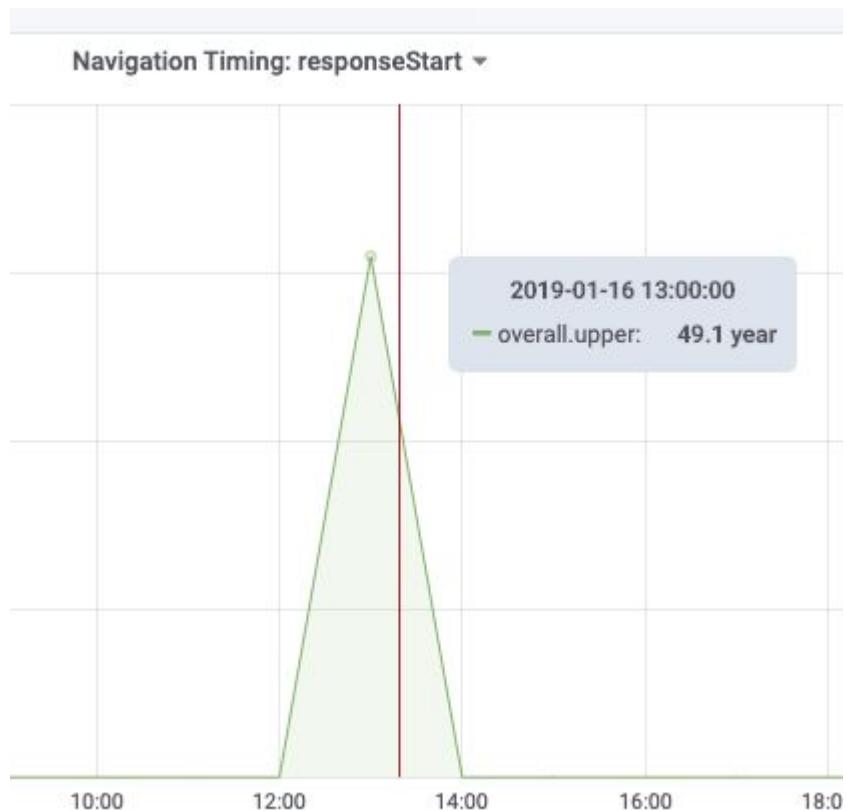
Nous avons mesuré la première image d'un article, et dans notre cas ça diffère de First Paint 25% du temps. Cependant la première image d'un article Wikipedia est parfois plus bas que l'écran initial. Attention, car dans ce cas la mesure peut différer en fonction du comportement de la personne.

En attendant la sortie d'Element Timing, Chrome a sorti il y a 10 jours le plus générique [Largest Contentful Paint](#).

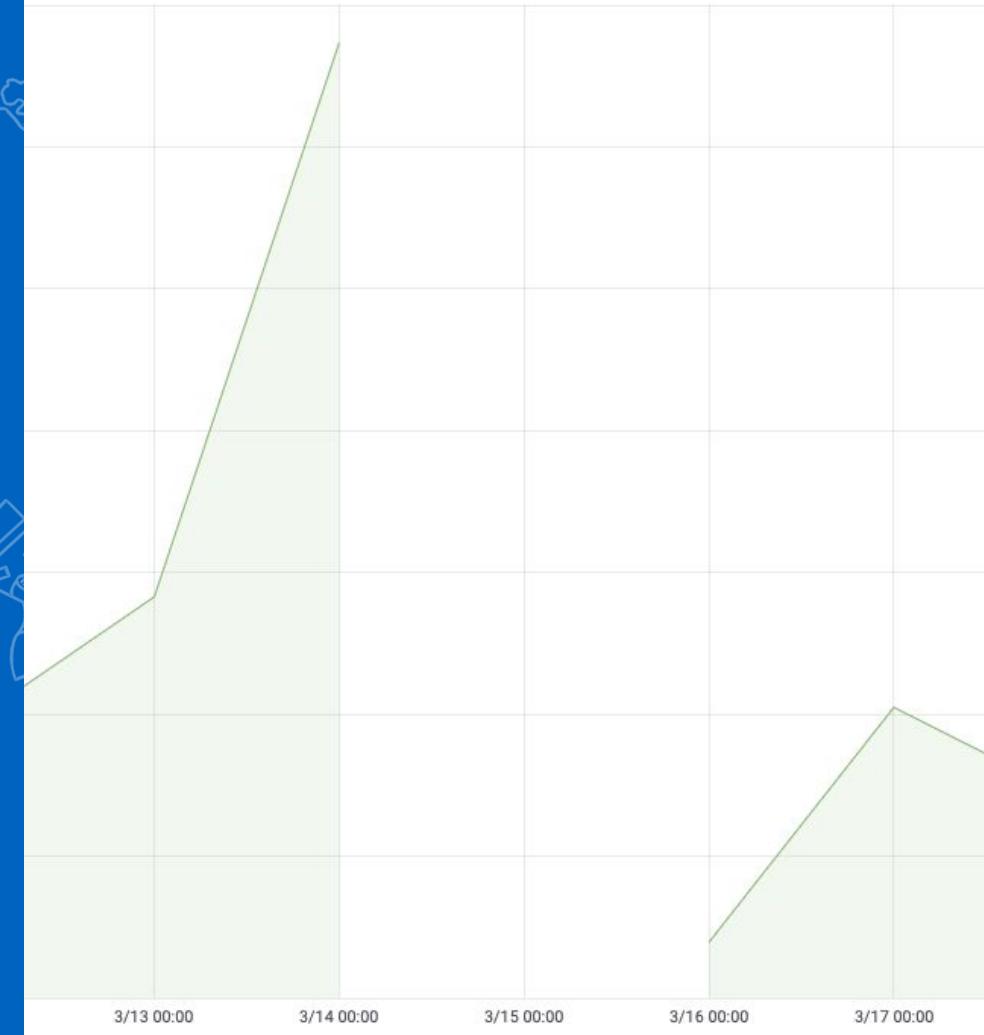
Quel est le problème?



**Les données
brutes sont
bruyantes et
peuvent avoir des
valeurs extrêmes**



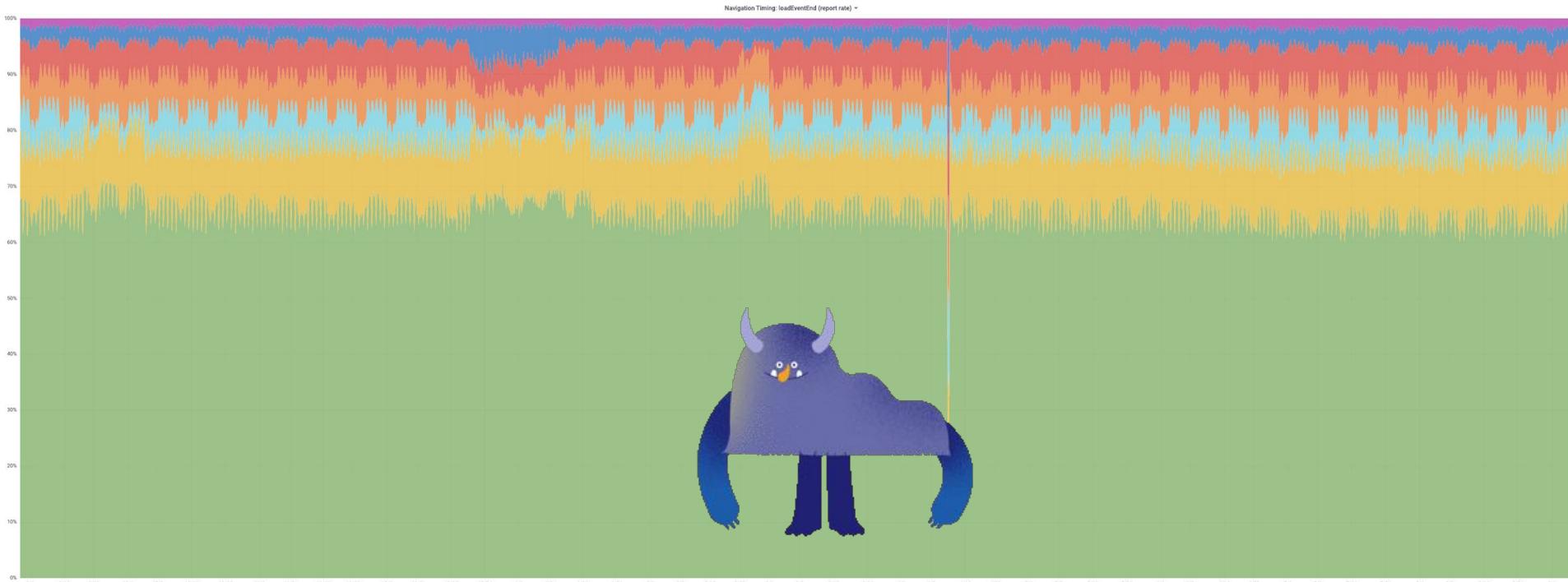
Il est difficile de mesurer les données RUM parfaitement sur le long terme



Chaque site a un public différent, donc une composition globale spécifique

Site	Part de marché Firefox	Score CPU médian	Mémoire disponible >= 4GO
es.wikipedia.org	12%	133	81%
es.m.wikipedia.org	0,5%	330	38%
en.wikipedia.org	14,1%	90	93,7%
fr.wikipedia.org	26%	109	93,4%

La composition évolue avec le temps



La corrélation de ces mesures avec la perception est faible

API	Mesure	Coefficient de corrélation Pearson
Navigation Timing	loadEventEnd	0.065
Navigation Timing	domInteractive	0.048
Paint Timing	first-paint	0.034
Network Information	RTT	0.077

Basé sur 180.000 mesures du site mobile es.m.wikipedia.org

On a peut-être trop focalisé sur le début de la construction de la page

Chargement initial de la page	L'expérience qui suit	Les deux
Navigation Timing (2010)	Long Tasks (2017)	User Timing (2013)
First Paint (2011)	Event Timing (?)	Resource Timing (2015)
First Contentful Paint (2017)		Element Timing (?)
Largest Contentful Paint (2019)		Layout Instability (?)

Quelques conseils

Sélectionnez les mesures les plus pertinentes

- Gardez une ou deux mesures “standard”, pour pouvoir vous comparer aux autres.
- Identifiez les aspects clefs de votre expérience utilisateur.
- Dans l’idéal, corrélez vos mesures de performances au comportement utilisateur, pour vérifier que vous mesurez quelque chose d’utile.
- Essayez de capturer des aspects différents de l’expérience, il est rapidement inutile de mesurer 10 événements proches dans les temps les uns des autres pendant l’affichage initial de la page.
- N’hésitez pas à demander à un échantillon d’utilisateurs de votre site ce qu’ils pensent de votre performance, les résultats pourraient vous surprendre.

Filtrez les données brutes

Un temps de chargement de page qui prend des années, ou qui est totalement instantané (0ms), voire négatif ne sont absolument pas réalistes et doivent être filtrés.

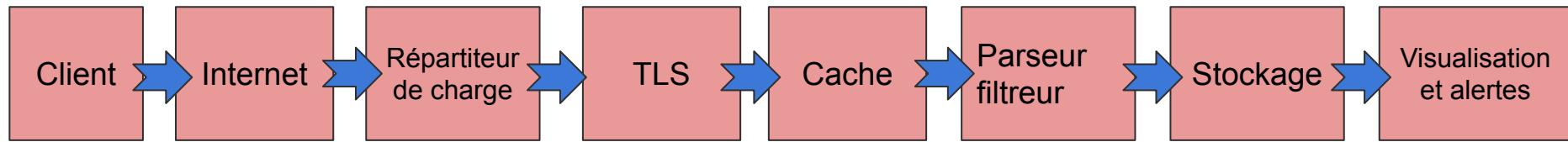
De même, les mesures sont faussées si l'onglet était initialement caché (donc “endormi”). On peut les filtrer grâce à [Page Visibility](#).

Nous avons également constaté des problèmes de fiabilité avec les mesures lors d'une page rafraîchie. Filtrable avec [Navigation Timing](#).

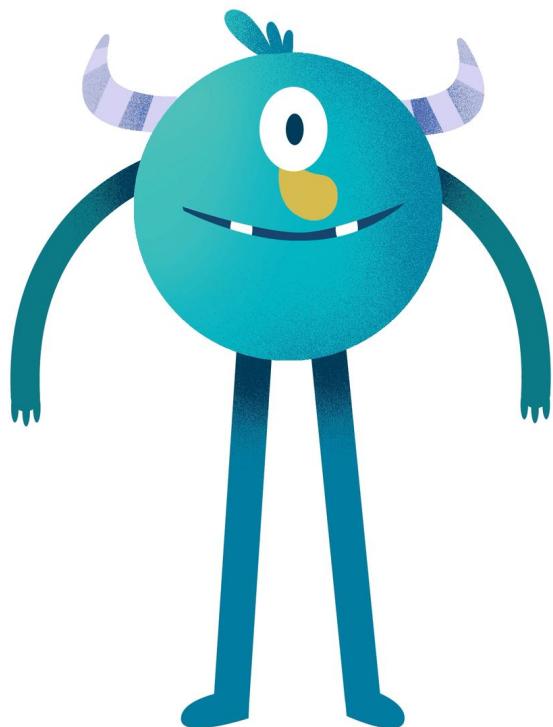
Remettez régulièrement en doute la chaîne d'approvisionnement



Remettez régulièrement en doute la chaîne d'approvisionnement



Zoomez, dézoomez



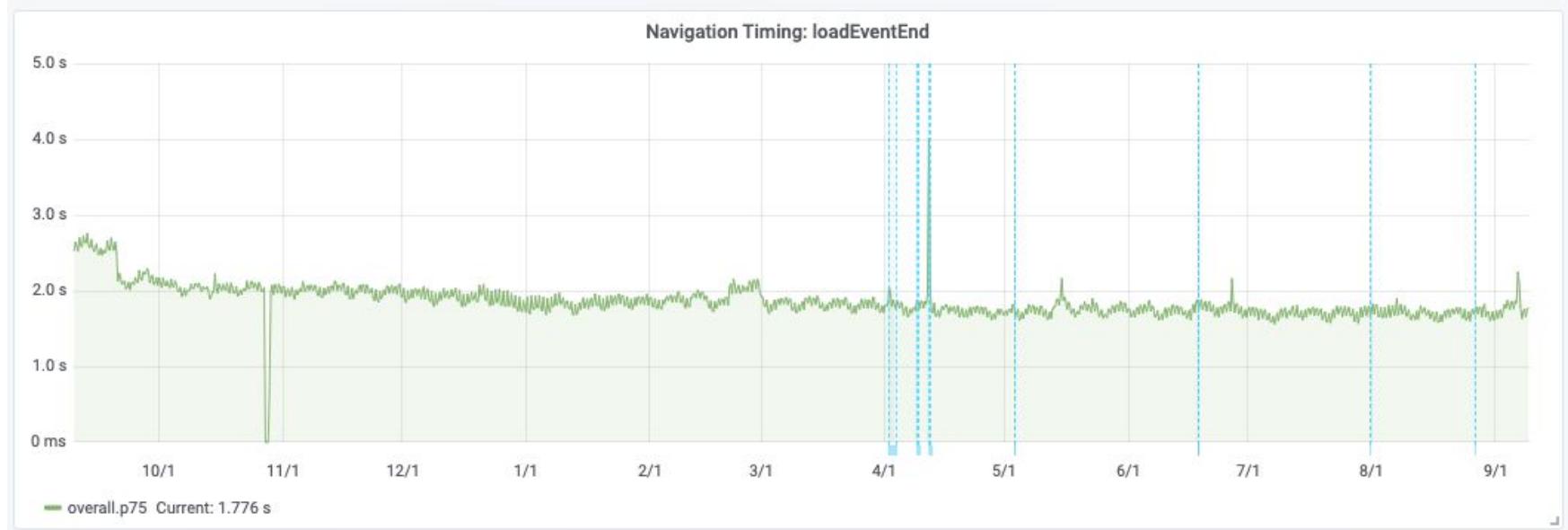
Zoomez, dézoomez

▼ Compared to last week



Zoomez, dézoomez

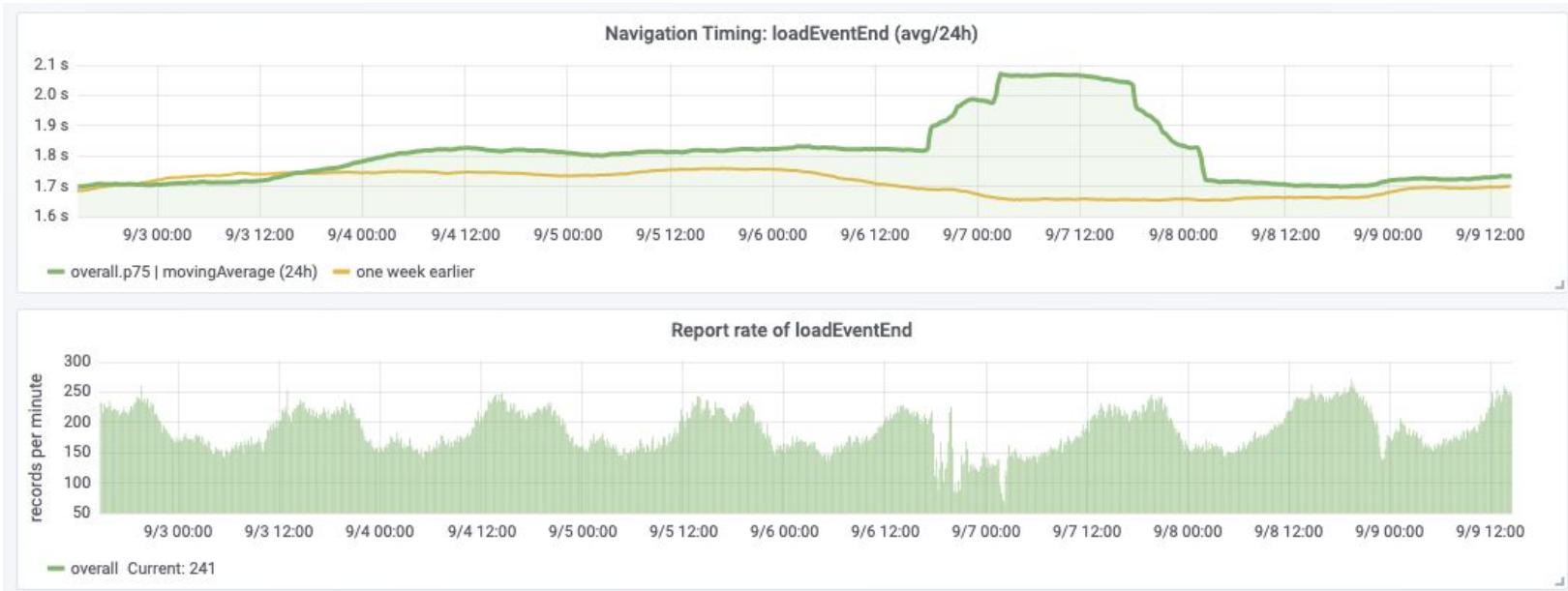
Overview



Regardez les données sous des angles différents



Regardez les données sous des angles différents



Regardez les données sous des angles différents



Observez par tranches

Exemple: visiteurs se connectant depuis la France, accédant au site mobile avec Safari et moins de 4Go de mémoire disponible.

Un site marchand pourrait également faire un choix moins arbitraire et se concentrer sur le type d'appareil et le navigateur le plus courant chez ses meilleurs clients.

Comparez-vous aux autres



Google CrUX & Pagespeed Insights





MERCI

(on recrute!)



WIKIMEDIA
FOUNDATION