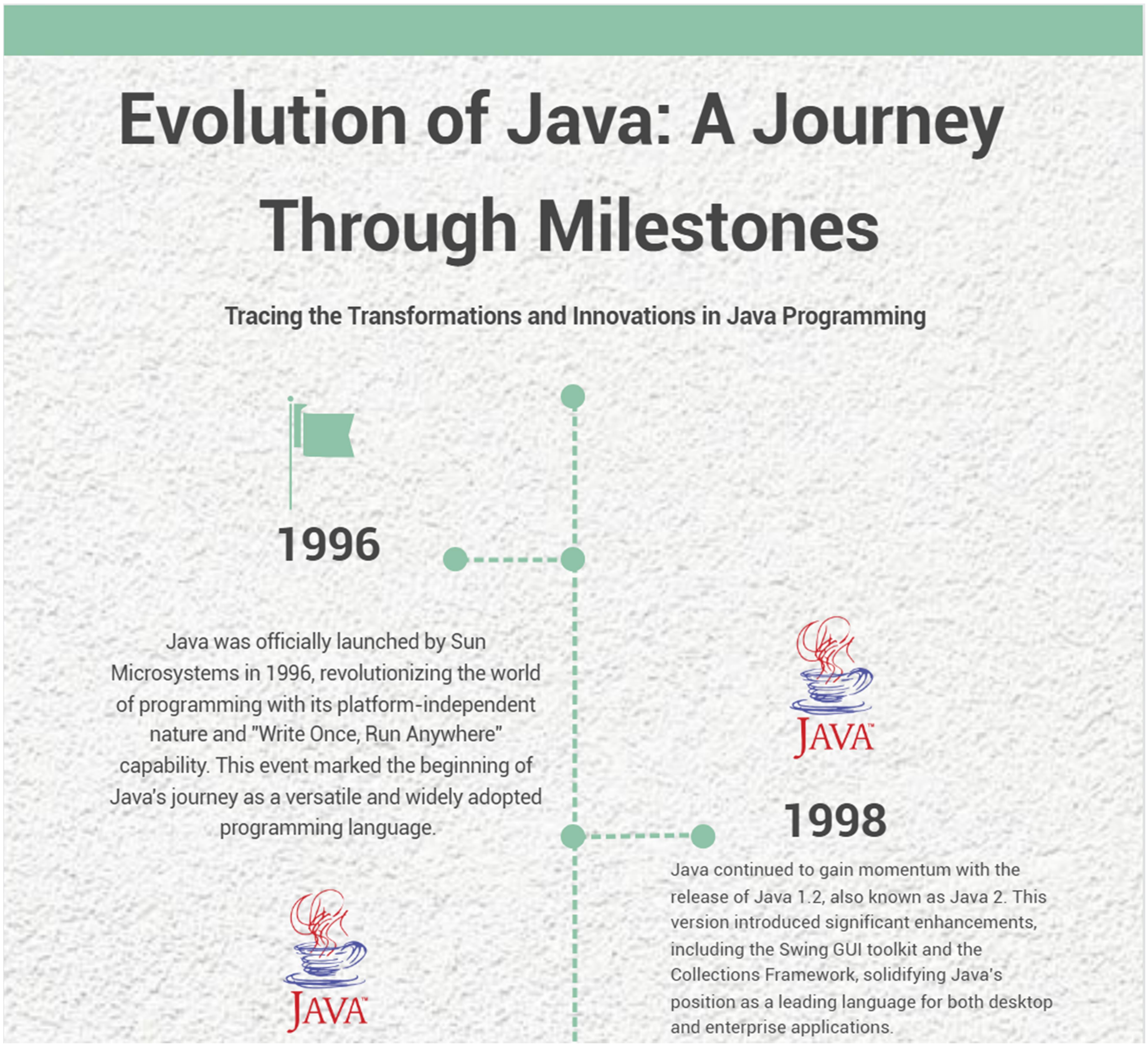
PRACTICAL - 4



Evolution of Java: A Journey Through Milestones

Tracing the Transformations and Innovations in Java Programming

1996

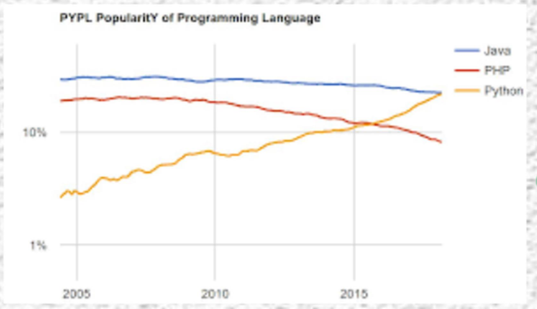
Java was officially launched by Sun Microsystems in 1996, revolutionizing the world of programming with its platform-independent nature and "Write Once, Run Anywhere" capability. This event marked the beginning of Java's journey as a versatile and widely adopted programming language.

1998

Java continued to gain momentum with the release of Java 1.2, also known as Java 2. This version introduced significant enhancements, including the Swing GUI toolkit and the Collections Framework, solidifying Java's position as a leading language for both desktop and enterprise applications.

2002

Java made substantial progress with the release of Java 1.4. This version introduced important features like the assert keyword for easier debugging, the Java Naming and Directory Interface (JNDI) for directory services, and improvements in performance, making Java an even more robust and capable programming platform.



2006

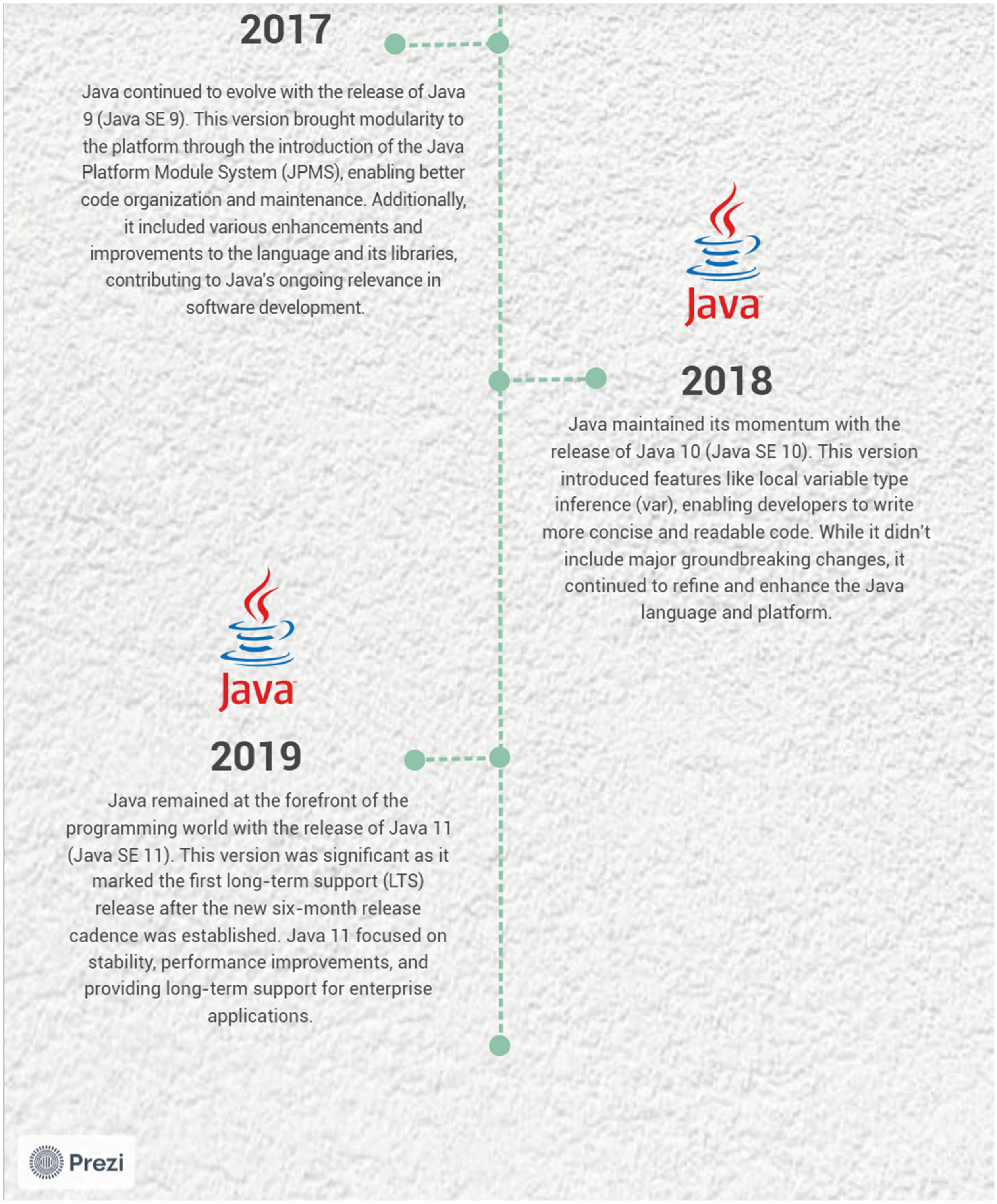
Java continued its evolution with the release of Java Platform, Standard Edition 6 (Java SE 6), also known as Java 6. This version brought significant improvements such as scripting support through the inclusion of the scripting API, enhanced performance, and expanded libraries, further cementing Java's role as a versatile and powerful programming language.

2011

Java underwent a major transformation with the release of Java 7 (Java SE 7). This version introduced features like the try-with-resources statement for improved resource management, the diamond operator for simplifying generic type instantiation, and the ForkJoinPool for concurrent programming, enhancing Java's capabilities for modern software development.

2014

Java took another leap forward with the release of Java 8 (Java SE 8). This version was a game-changer, introducing lambdas, the Stream API, and the java.time package for modern date and time handling. These additions revolutionized Java's approach to functional programming and made it even more suitable for contemporary software development.



2017

Java continued to evolve with the release of Java 9 (Java SE 9). This version brought modularity to the platform through the introduction of the Java Platform Module System (JPMS), enabling better code organization and maintenance. Additionally, it included various enhancements and improvements to the language and its libraries, contributing to Java's ongoing relevance in software development.

2018

Java maintained its momentum with the release of Java 10 (Java SE 10). This version introduced features like local variable type inference (var), enabling developers to write more concise and readable code. While it didn't include major groundbreaking changes, it continued to refine and enhance the Java language and platform.

2019

Java remained at the forefront of the programming world with the release of Java 11 (Java SE 11). This version was significant as it marked the first long-term support (LTS) release after the new six-month release cadence was established. Java 11 focused on stability, performance improvements, and providing long-term support for enterprise applications.

Signature of Faculty: ___________ Date: _________